翼 MapReduce 服务（MRS）

故障排除

天翼云科技有限公司

目 录

1 Web 页面访问类

1.1 无法访问 MRS 集群管理页面（MRS Manager 界面）

问题现象

集群创建完成后，无法访问集群管理页面，即无法访问 MRS Manager 界面。

原因分析

- 需要对 MRS 节点绑定弹性 IP 才可访问
- 需要添加安全组规则，放开 9022 端口

处理步骤

步骤 1 登录 MRS 的 Console 页面，在现有集群列表中找到需要访问的集群，单击集群名称。

步骤 2 在节点信息中单击需要访问的节点名称，选择“弹性公网 IP” > “绑定弹性公网 IP”。

步骤 3 在“绑定弹性公网”IP 页面，“选择网卡”下拉框中选择需要绑定的网卡，“选择弹性公网 IP”中选择需要绑定的弹性公网 IP，单击“确定”。

步骤 4 弹性 IP 绑定成功后，需要将安全组规则中 9022 端口放开。

选择“安全组”页签，单击“更改安全组”。

可以选择添加已有的安全组，或者单击“新建安全组”，进入安全组管理界面进行创建，添加用户访问公网 IP 地址 9022 端口的安全组规则。

步骤 5 添加成功后，可通过 *https://弹性 ip:9022/mrsmanager/*访问 MRS。如果还未能访问，请联系技术支持。

----结束

1.2 升级 Python 后，无法登录 MRS Manager 页面

用户问题

升级 Python 后，登录不进去 MRS Manager 页面。

问题现象

自行升级 Python 后，使用 admin 帐号且密码正确的情况下登录不进去 MRS Manager 页面。

原因分析

用户升级 Python 版本到 Python3.x 的过程中，修改了 openssl 的文件目录权限，导致 LdapServer 服务无法正常启动，从而引起登录认证失败。

处理步骤

步骤 1 以 root 用户登录集群的 Master 节点。

步骤 2 执行 **chmod 755 /usr/bin/openssl** 命令，修改**/usr/bin/openssl** 的文件目录权限为 755。

步骤 3 执行 **su omm** 命令，切换到 omm 用户。

步骤 4 执行 **openssl** 命令，查看是否能够进入 openssl 模式。

如果能够成功进入，则表示权限修改成功，如果不能进入，则表示权限未修改成功。

如果权限未修改成功，请检查执行的命令是否正确，或者联系运维人员。

步骤 5 权限修改成功后会重启 LdapServer 服务，请等待 LdapServer 服务重启成功后，重新登录 MRS Manager。

----结束

建议与总结

自行安装的软件建议和系统的分开，系统软件升级可能造成兼容性问题。

1.3 用户修改域名后无法登录 MRS Manager 页面

问题现象

用户修改域名后，通过 console 页面无法登录 MRS Manager 页面，或者登录 MRS Manager 页面异常。

问题原因

用户修改域名后，没有刷新 executor 用户的 keytab 文件，导致 executor 进程认证失败后不断循环认证，导致了 acs 进程内存溢出。

解决方案

步骤 1 重启 acs 进程。

1. 使用 root 用户登录主管理节点（即 MRS 集群详情页面“节点管理”页签下实心五角星所在的 Master 节点）。
2. 执行如下命令重启进程：

   su - omm

   ps -ef|grep =acs （查找 acs 进程 PID）

   kill -9 *PID* （PID 替换为实际的 ID，结束 acs 进程）
3. 等待几分钟后执行命令 **ps -ef|grep =acs** 查询进程是否已经自动启动。

步骤 2 替换 executor 用户的 keytab 文件。

1. 登录 MRS Manager 页面，选择“系统 > 用户”，在 executor 用户所在的“操作”列，单击“下载认证凭据”，解压后获取 keytab 文件。
2. 使用 root 用户登录主管理节点，将获取的 keytab 替换“/opt/executor/webapps/executor/WEB-INF/classes/user.keytab”文件。

步骤 3 替换 knox 用户的 keytab 和 conf 文件。

1. 登录 MRS Manager 页面，选择“系统 > 用户”，在 knox 用户所在的“操作”列，单击“下载认证凭据”，解压后获取 keytab 和 conf 文件。
2. 使用 root 用户登录主管理节点，将获取的 keytab 替换“/opt/knox/conf/user.keytab”文件。
3. 修改/opt/knox/conf/krb5JAASLogin.conf 中的 principal 的值，把域名修改为更改后的域名。
4. 将获取的 krb5.conf 替换“/opt/knox/conf/krb5.conf”文件。

步骤 4 备份原有客户端目录

mv *{客户端目录}* /opt/client_init

步骤 5 重新安装客户端。

步骤 6 使用 root 用户登录主备管理节点，执行如下命令，重启 knox 进程。

su - omm

ps -ef | grep gateway | grep -v grep （查找 knox 进程 PID）

kill -9 *PID*（PID 替换为实际的 ID，结束 knox 进程）

/opt/knox/bin/restart-knox.sh （启动 knox 进程）

步骤 7 使用 root 用户登录主备管理节点，执行如下命令，重启 executor 进程。

su - omm

netstat -anp |grep 8181 |grep LISTEN （查找 executor 进程 PID）

kill -9 *PID*（PID 替换为实际的 ID，结束 executor 进程）

/opt/executor/bin/startup.sh （启动 executor 进程）

----结束

1.4 登录 Manager，页面空白不显示

用户问题

登录到 FusionInsight Manager 界面后，页面空白不显示。

问题现象

登录到 FusionInsight Manager 界面后，页面空白不显示。

原因分析

Manager 无法登录，需要清除浏览器缓存。

处理步骤

步骤 1 切换至浏览器窗口（以 Chrome 为例），通过键盘按下“Ctrl+Shift+Delete”弹出“清除浏览数据”对话框。

步骤 2 勾选待清除的浏览记录，单击“清除数据”，完成浏览器缓存清理。

----结束

2 集群管理类

2.1 缩容 Task 节点失败

用户问题

客户在 MRS 2.x 集群详情界面执行调整集群，将 Task 节点调整成 0 个，最终缩容失败。

问题现象

客户在 MRS 集群详情页面调整集群 Task 节点，最终缩容失败，提示“This operation is not allowed because the number of instances of NodeManager will be less than the minimum configuration after scale-in, which may cause data loss.”

原因分析

客户将 Core 节点的 NodeManager 服务停止了，导致在检查 Task 节点退服过程中发现 Task 如果全部退订，则将没有 NodeManager，则 Yarn 服务就不可用，而 MRS 判断剩余的 NodeManger 必须大于等于 1 才能退服 Task 节点。

处理步骤

步骤 1 勾选 Core 节点的 NodeManager 实例，选择“更多 > 启动实例”。

步骤 2 在集群列表页面缩容 Task 节点。

1. 单击集群名称进入集群详情页面，选择“节点管理”。
2. 在 Task 节点组所在行的“操作”列单击“缩容”。
3. 单击“确定”并在弹出框选择“是”。

步骤 3 等缩容成功后，若不想用 Core 节点的 NodeManager 再将其停止。

----结束

建议与总结

Core 节点的 NodeManager 通常不会将其停止，客户不要随意变更集群部署结构。

2.2 如何处理集群内部 OBS 证书过期

用户问题

用户在 MRS 集群中访问 OBS 服务过程中出现证书过期问题。

问题现象

MRS 集群产生“ALM-12054 证书文件失效”或“ALM-12055 证书文件即将过期”告警，且告警详情中触发告警的证书为 OBS 证书。

图2-1 OBS 证书即将过期告警

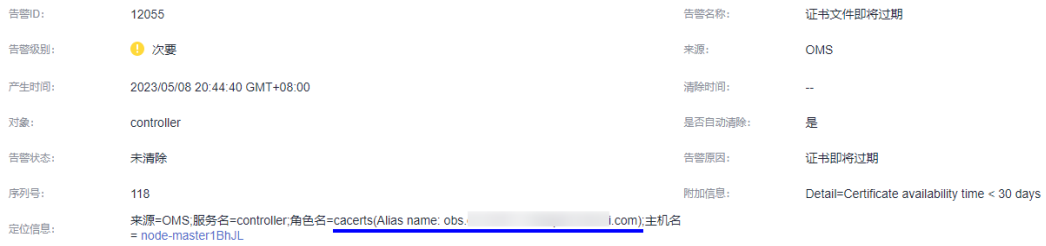


图2-2 OBS 证书失效告警

原因分析

OBS 系统生成的证书有有效期限制，到达有效期后，证书文件失效，因此产生告警。

用户导入新的有效证书后，服务器端会自动更新证书，且对集群无影响。

处理步骤

步骤 1 查询 MRS 集群上的 OBS 证书信息。

使用 **root** 用户登录 MRS 集群的主 OMS 节点，执行以下命令查询是否存在 OBS 相关证书。

keytool -v -list -keystore *${java_home}*/**jre/lib/security/cacerts -protected 2> /dev/null|grep -E "Alias name*|Valid from*" | grep obs**

如下示例表示存在 OBS 证书：

```
Alias name: obs.xxx.com
```

- 若不存在证书，则无需进行后续操作，等待告警恢复。
- 若存在证书，则执行步骤 2。

说明

- *${java_home}*为集群 JDK 目录，MRS 3.x 版本中，替换为“/opt/Bigdata/common/runtime0/jdk1.8*”，MRS 3.x 之前版本中，替换为“/opt/Bigdata/jdk”。
- MRS 3.x 版本集群若按照该指导执行后依然出现证书过期告警，请将*${java_home}*替换为“/opt/Bigdata/client/JDK/jdk”，再次执行该指导。

步骤 2 删除 OBS 证书。

在主 OMS 节点中，执行以下命令删除步骤 1 中查询到的 OBS 证书。

obs_url=$(keytool -v -list -keystore *${java_home}*/jre/lib/security/cacerts -protected 2> /dev/null|grep -E "Alias name*|Valid from*" | grep obs | cut -d ':' -f 2 | awk '$1=$1')

jdk_cacert="*${java_home}*/jre/lib/security/cacerts"

keytool -delete -alias ${obs_url} -keystore ${jdk_cacert} -storepass changeit

步骤 3 执行以下命令，确认 OBS 证书已不存在，若仍存在，则继续执行步骤 2。

keytool -v -list -keystore *${java_home}*/jre/lib/security/cacerts -protected 2> /dev/null|grep -E "Alias name*|Valid from*" | grep obs

----结束

2.3 MRS 集群添加新磁盘

用户问题

MRS HBase 服务不可用。

问题现象

用户主机的磁盘占用率过高导致服务故障。

原因分析

Core 节点的磁盘容量不足导致无法提供正常服务。

处理步骤

⚠ 注意

如果购买 MRS 集群的计费模式为按需计费，扩容磁盘容量后 MRS 集群不支持转包周期。

步骤 1 购买云硬盘。

步骤 2 挂载云硬盘。

- 若挂载云硬盘完成，请执行步骤 6。
- 若在云硬盘控制台执行 "挂载" 操作时无法选定云服务器，请执行步骤 3。

步骤 3 登录弹性云主机控制台，单击待扩容（挂载新磁盘）的弹性云主机名称。

步骤 4 在“云硬盘”页签，单击“挂载磁盘”。

步骤 5 选择待挂载的新磁盘并单击“确定”完成磁盘挂载。

步骤 6 初始化 Linux 数据盘。

说明

- 挂载点目录根据节点 DataNode 已有的实例编号递增，例如：使用 df –h 命令查到当前已有的编号为/srv/BigData/hadoop/data1，则新增挂载点为/srv/BigData/hadoop/data2。初始化 Linux 数据盘新建挂载点时，将新建挂载点命名为/srv/BigData/hadoop/data2，并将新建分区挂载到该挂载点下。例如

```
mkdir /srv/BigData/hadoop/data2
mount /dev/xvdb1 /srv/BigData/hadoop/data2
```

/srv/BigData/hadoop/data2 路径说明：本章节后续提到/srv/BigData/hadoop/data2 路径均请按照以下场景自行修改。

- 3.X 版本目录为：/srv/BigData/data2
- 3.X 之前版本目录为：/srv/BigData/hadoop/data2

步骤 7 执行以下命令为新磁盘增加 omm 用户权限。

chown omm:wheel *新增挂载点*

例如：**chown omm:wheel /srv/BigData/hadoop/data2**

步骤 8 执行 chmod 701 命令为新增的挂载点目录添加执行权限。

chmod 701 *新增挂载点*

例如：**chmod 701 /srv/BigData/hadoop/data2**

说明

chmod 701 命令中 701 仅为示例，请以已有数据盘 data1 的数值为准。

步骤 9 登录 Manager，扩容 DataNode 实例和 NodeManager 实例的数据磁盘。

步骤 10 修改当前节点 DataNode 实例配置。

MRS Manager 界面操作入口：登录 MRS Manager，依次选择 “服务管理 > HDFS > 实例 > 扩容的 DataNode 节点 > 实例配置”，“参数类别”选择 "全部配置"。

FusionInsight Manager 界面操作入口：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > HDFS > 实例 > 扩容的 DataNode 节点 > 实例配置”，选择 "全部配置"。

- 方式一：手动修改当前节点 DataNode 实例配置。
  - 在 "搜索" 中输入"dfs.datanode.fsdataset.volume.choosing.policy"，将参数值改为 "org.apache.hadoop.hdfs.server.datanode.fsdataset.AvailableSpaceVolumeChoosingPolicy"。
  - 在 "搜索" 中输入 "dfs.datanode.data.dir"，将参数值改为 "/srv/BigData/hadoop/data1/dn,/srv/BigData/hadoop/data2/dn"

  若此两个参数有修改，则单击 "保存配置"，并勾选 "重启角色实例" ，重启 DataNode 实例。
- 方式二：自动同步当前节点 DataNode 实例配置。
  a. 单击“同步配置”为 HDFS 服务启用新的配置参数。
  b. 完成同步配置后，请重启实例以使配置生效。

说明

- 如果确认当前未使用 HDFS，并且希望较快完成重启，可以选择直接"重启角色实例" 。
- 如果有任务在使用 HDFS，为了防止数据异常或者任务失败，必须选择滚动重启。

步骤 11 修改当前节点 Yarn NodeManager 的实例配置。

MRS Manager 界面操作入口：登录 MRS Manager，依次选择 "服务管理 > Yarn > 实例 > 扩容节点的 NodeManager > 实例配置"，"参数类别" 选择 "全部配置"。

FusionInsight Manager 界面操作入口：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Yarn > 实例”单击扩容节点的 NodeManager，选择“实例配置 > 全部配置”。

- 方式一：手动修改当前节点 Yarn NodeManager 的实例配置。
  - 在“搜索”中输入 "yarn.nodemanager.local-dirs"，将参数值修改为："/srv/BigData/hadoop/data1/nm/localdir,/srv/BigData/hadoop/data2/nm/localdir" 。
  - 在“搜索”中输入 "yarn.nodemanager.log-dirs"，将参数值修改为："/srv/BigData/hadoop/data1/nm/containerlogs,/srv/BigData/hadoop/data2/nm/containerlogs"。

    若此两个参数有修改，则保存配置，并勾选 "重启角色实例" ，重启 NodeManager 实例。
- 方式二：自动同步当前节点 Yarn NodeManager 的实例配置。
  a. 单击“同步配置”为 Yarn 服务启用新的配置参数。
  b. 完成同步配置后，请重启实例以使配置生效。

说明

- 如果确认当前未使用 Yarn，并且希望较快完成重启，可以选择直接"重启角色实例" 。

- 如果有任务在使用 Yarn，为了防止数据异常或者任务失败，必须选择滚动重启。

步骤 12 查看扩容是否成功。

MRS Manager 界面操作：登录 MRS Manager，依次选择 "服务管理 > HDFS > 实例 > 扩容的 DataNode 节点 "。

FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > HDFS > 实例”，单击扩容的 DataNode 节点。

在图表区域，查看实时监控项 "DataNode 存储" 中 配置的总磁盘容量是否提升，若图表区域没有监控项 "DataNode 存储" ，请单击“定制”增加该监控项。

- 若配置的总磁盘容量已提升，则扩容完成。
- 若配置的总磁盘容量未提升，请联系技术支持处理。

步骤 13 （可选）扩容 Kafka 实例的数据盘。

修改当前节点 Kafka 实例配置。

1. 进入 Kafka 扩容的 Broker 节点参数配置界面。

   MRS Manager 界面操作：登录 MRS Manager，依次选择 "服务管理 > Kafka > 实例 > 扩容的 Broker 节点 > 实例配置"，"参数类别" 选择 "全部配置"。

   FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka > 实例> 扩容的 Broker 节点 > 实例配置”， 选择“全部配置”。

2. 在 "搜索" 中输入"log.dirs"，加入新增磁盘信息，中间用英文 “,” 分割。

   例如原始只有一块 Kafka 数据盘，新增一块，则将"/srv/BigData/kafka/data1/kafka-logs" 改为 "/srv/BigData/kafka/data1/kafka-logs,/srv/BigData/kafka/data2/kafka-logs"。

3. 保存配置，并勾选 "重启角色实例" 后根据提示重启实例。

4. 查看扩容是否成功。

   MRS Manager 界面操作入口：登录 MRS Manager，依次选择 "服务管理 > Kafka > 实例 > 扩容的 Broker 节点 "。

   FusionInsight Manager 界面操作入口：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka > 实例> 扩容的 Broker 节点 ”。

   查看实时监控项 "Broker 磁盘容量大小" 中配置的总磁盘容量是否提升。

步骤 14 设置开机自动挂载磁盘分区。

----结束

须知

集群的节点扩容磁盘之后，若再扩容集群节点时需要在新扩容的节点上参考该页面处理步骤执行添加磁盘的操作，否则会有数据丢失的风险 。

建议与总结

- 当磁盘的使用率超过 85%时，建议用户进行磁盘扩容，并将新购买的磁盘挂载到弹性云主机上与集群进行关联。
- 具体挂载步骤、配置参数请根据实际情况进行。

2.4 MRS 集群更换磁盘（适用于 2.x 及之前）

用户问题

磁盘无法访问。

问题现象

客户创建本地盘系列 MRS 集群，其中 1 个 Core 节点的磁盘存在硬件损坏，导致读取文件失败。

原因分析

磁盘硬件故障。

处理步骤

说明

该指导适用于 MRS 3.x 之前版本分析集群，如需为流式集群或混合集群更换磁盘，请联系技术支持处理。

步骤 1 登录。

步骤 2 选择“主机管理”并单击需要退服主机的“主机名称”，在“角色”列表中单击 RegionServer，选择“更多 > 退服”。

步骤 3 选择“主机管理”并单击需要退服主机的“主机名称”，在“角色”列表中单击 DataNode，选择“更多 > 退服”。

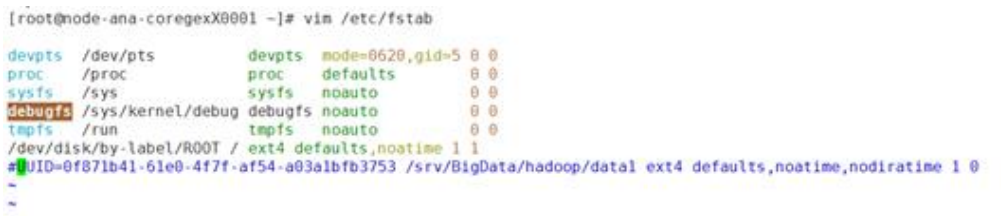
步骤 4 选择“主机管理”并单击需要退服主机的“主机名称”，在“角色”列表中单击 NodeManager，选择“更多 > 退服”。

说明

该主机下若还有其他实例，请参考该步骤方式进行退服。

步骤 5 执行 **vim /etc/fstab** 命令编辑注释旧磁盘的挂载点。

图2-3 注释旧磁盘的挂载点



步骤 6 迁移旧磁盘上（例如：/srv/BigData/hadoop/data1/）的用户自有数据。

步骤 7 登录 MRS 管理控制台。

步骤 8 在集群详情页面，选择“节点管理”。

步骤 9 单击待更换磁盘的“节点名称”进入弹性云主机管理控制台，单击“关机”。

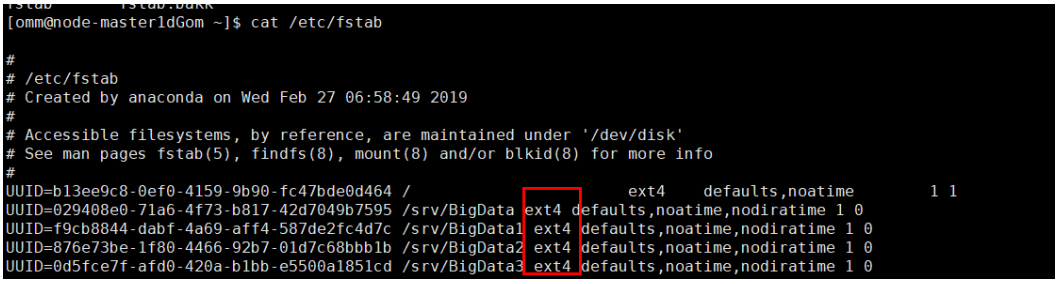
步骤 10 联系支持人员在后台更换磁盘。

步骤 11 在弹性云主机管理控制台，单击“开机”，将已更换磁盘的节点开机。

步骤 12 执行 **fdisk -l** 命令，查看新增磁盘。

步骤 13 使用 **cat /etc/fstab** 获取盘符。

图2-4 获取盘符



步骤 14 使用对应的盘符对新磁盘进行格式化。

例如：**mkfs.ext4 /dev/sdh**

步骤 15 执行如下命令挂载新磁盘。

mount ***新磁盘 挂载点***

例如：**mount /dev/sdh /srv/BigData/hadoop/data1**

步骤 16 执行如下命令为新磁盘增加 omm 用户权限。

chown omm:wheel ***挂载点***

例如：**chown -R omm:wheel /srv/BigData/hadoop/data1**

步骤 17 在 fstab 文件中新增新磁盘 UUID 信息。

1. 使用 **blkid** 命令查看新磁盘的 UUID。

```
[root@node-ana-coreKpoT0003 ~]# blkid
/dev/vda1: LABEL="ROOT" UUID="2aa97872-11ec-422e-9513-0f28b925ad5e" TYPE="ext4"
/dev/vdb: UUID="e5f652c3-f9af-427f-89da-f2545618688d" TYPE="ext4"
[root@node-ana-coreKpoT0003 ~]#
```

2. 打开“/etc/fstab”文件，新增如下信息：

```
UUID=新盘 UUID /srv/BigData/hadoop/data1 ext4 defaults,noatime,nodiratime 1 0
```

步骤 18 （可选）执行如下命令新建日志目录。

mkdir -p /srv/BigData/Bigdata

chown omm:ficommon /srv/BigData/Bigdata

chmod 770 /srv/BigData/Bigdata

说明

执行如下命令确认 Bigdata 日志软链接目录是否已存在，若存在则忽略该步骤。

ll /var/log

步骤 19 登录。

步骤 20 选择“主机管理”并单击需要入服主机的“主机名称”，在“角色”列表中单击 RegionServer，选择“更多 > 入服”。

步骤 21 选择“主机管理”并单击需要入服主机的“主机名称”，在“角色”列表中单击 DataNode，选择“更多 > 入服”。

步骤 22 选择“主机管理”并单击需要入服主机的“主机名称”，在“角色”列表中单击 NodeManager，选择“更多 > 入服”。

说明

该主机下若还有其他实例，请参考该步骤方式进行入服。

步骤 23 选择“服务管理 > HDFS”，在“服务状态”页签的“HDFS 概述”模块查看“丢失块数”是否为“0”。

- “丢失块数”是为“0”，则操作完成。
- “丢失块数”不为“0”，请联系支持人员进行处理。

----结束

2.5 MRS 集群更换磁盘（适用于 3.x）

用户问题

磁盘无法访问。

问题现象

客户创建本地盘系列 MRS 集群，其中 1 个 Core 节点的磁盘存在硬件损坏，导致读取文件失败。

原因分析

磁盘硬件故障。

处理步骤

说明

该指导适用于本地盘系列（d/i/ir/ki 系列）MRS 集群，针对 Core、Task 类型节点的磁盘存在硬件故障。

Kafka 组件不支持更换磁盘，如果存储 Kafka 数据的节点故障，请联系技术支持处理。

步骤 1 登录。

步骤 2 选择“主机”并单击故障主机的“主机名称”，在“实例”列表中单击 DataNode，选择“更多 > 退服”。

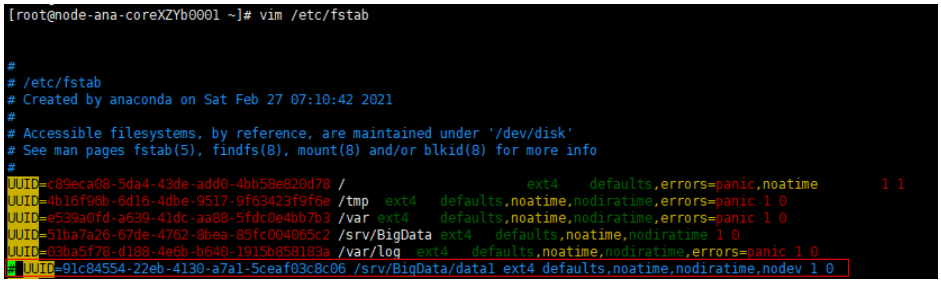
说明

- 该主机下若存在 DataNode、NodeManager、RegionServer 和 ClickHouseServer 实例，请参考该步骤进行退服操作；
- MRS 3.1.2 版本之后支持退服 ClickHouseServer 角色实例。

步骤 3 选择“主机”并勾选故障主机“主机名称”前的复选框，选择“更多 > 停止所有实例”。

步骤 4 执行 **vim /etc/fstab** 命令编辑注释旧磁盘的挂载点。

图2-5 注释旧磁盘的挂载点



步骤 5 如果旧磁盘仍可访问，迁移旧磁盘上（例如：/srv/BigData/data1/）的用户自有数据。

cp -r ***旧磁盘挂载点 临时数据保存目录***

例如：**cp -r /srv/BigData/data1 /tmp/**

步骤 6 登录 MRS 管理控制台。

步骤 7 在集群详情页面，选择“节点管理”。

步骤 8 单击待更换磁盘的“节点名称”进入弹性云主机管理控制台，单击“关机”。

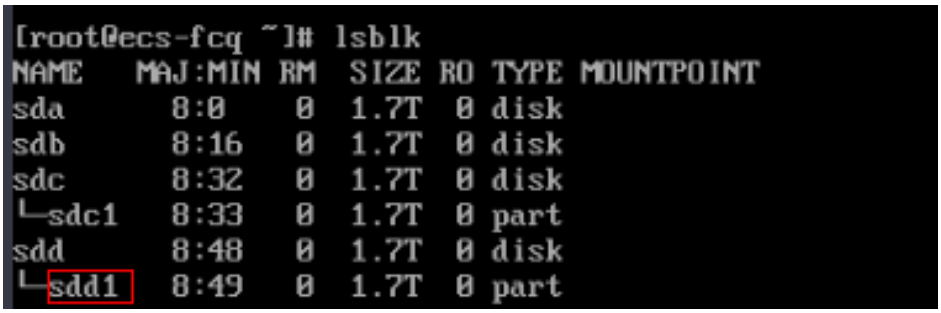
步骤 9 联系支持人员在后台更换磁盘。

步骤 10　在弹性云主机管理控制台，单击“开机”，将已更换磁盘的节点开机。

步骤 11　初始化 Linux 数据盘。

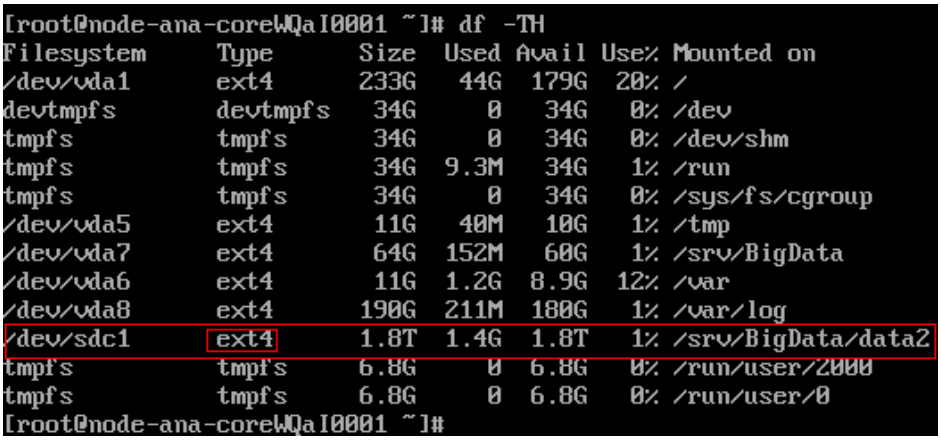
步骤 12　执行 **lsblk** 命令，查看新增磁盘分区信息。

图2-6 查看新增磁盘（分区）



步骤 13　使用 **df -TH** 获取文件系统类型。

图2-7 获取文件系统类型



步骤 14　使用对应的文件系统类型对新磁盘（分区）进行格式化。

例如：**mkfs.ext4 /dev/sdd1**

步骤 15　执行如下命令挂载新磁盘。

mount ***新磁盘 挂载点***

例如：**mount /dev/sdd1 /srv/BigData/data1**

说明

如果挂载不上，请执行如下命令重载配置后重新挂载。

systemctl daemon-reload

步骤 16　执行如下命令为新磁盘增加 omm 用户权限。

chown omm:wheel ***挂载点***

例如：**chown -R omm:wheel /srv/BigData/data1**

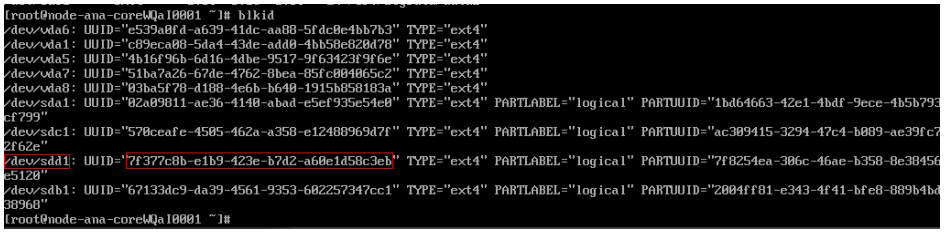
步骤 17 将旧磁盘上（例如：/srv/BigData/data1/）的用户自有数据迁移到新磁盘上。

cp -r ***临时数据保存目录 新磁盘挂载点***

例如：**cp -r /tmp/data1/* /srv/BigData/data1/**

步骤 18 在 fstab 文件中新增新磁盘 UUID 信息。

1. 使用 **blkid** 命令查看新磁盘的 UUID。



2. 打开“/etc/fstab”文件，新增如下信息：

```
UUID=新盘 UUID /srv/BigData/data1 ext4 defaults,noatime,nodiratime,nodev 1 0
```

步骤 19 登录。

步骤 20 选择“主机”并单击需要入服主机的“主机名称”，在“实例”列表中单击 DataNode，选择“更多 > 入服”。

说明

- 该主机下若存在 DataNode、NodeManager、RegionServer 和 ClickHouseServer 实例，请参考该步骤进行入服操作；
- MRS 3.1.2 版本之后支持入服 ClickHouseServer 角色实例。

步骤 21 选择“主机”，并勾选故障主机“主机名称”前的复选框，选择“更多 > 启动所有实例”。

步骤 22 选择“集群 > HDFS”，在“概览”页签的“基本信息”模块查看“丢失块数”是否为“0”。

- “丢失块数”是为“0”，则操作完成。
- “丢失块数”不为“0”，请联系支持人员进行处理。

----结束

2.6 MRS 备份失败

用户问题

MRS 备份总是失败。

问题现象

MRS 备份总是失败。

原因分析

备份目录软链接到系统盘，系统盘满了之后备份便会失败。

处理步骤

步骤 1 检查备份目录是否软链接到系统盘。

1. 以 root 用户登录集群主备 Master 节点。
2. 执行 **df -h** 命令查看磁盘情况，检查系统盘的存储情况。
3. 执行 **ll /srv/BigData/LocalBackup** 命令， 查看备份目录是否软连接到 **/opt/Bigdata/LocalBackup**。

   检查备份文件是否软链接到系统盘且系统盘空间是否足够。如果软链接到系统盘且系统盘空间不足，请执行步骤 2。如果否，说明不是由于系统盘空间不足导致，请联系技术服务。

步骤 2 将历史备份数据移到数据盘的新目录中。

1. 以 root 用户登录 Master 节点。
2. 执行 **su - omm** 命令，切换到 omm 用户。
3. 执行 **rm -rf /srv/BigData/LocalBackup** 命令，删除备份目录软连接。
4. 执行 **mkdir -p /srv/BigData/LocalBackup** 命令，创建备份目录。
5. 执行 **mv /opt/Bigdata/LocalBackup/* /srv/BigData/LocalBackup/**命令，将历史备份数据移到新目录。

----结束

2.7 Core 节点出现 df 显示的容量和 du 显示的容量不一致

用户问题

Core 节点出现 df 显示的容量和 du 显示的容量不一致

问题现象

Core 节点出现 df 显示的容量和 du 显示的容量不一致：

分别使用命令 **df -h** 和命令 **du -sh /srv/BigData/hadoop/data1/**查询得到的 /srv/BigData/hadoop/data1/目录磁盘占用量相差较大（大于 10G）。

原因分析

使用命令 **lsof |grep deleted** 可以查询到此目录下有大量 log 文件处于 deleted 状态。

出现此问题的一种情况是长时间运行某些 spark 任务，任务中的一些 container 一直运行，并且持续产生日志；spark 的 executor 在打印日志的时候使用了 log4j 的日志滚动功能，将日志输出到 stdout 文件下；而 container 同时也会监控这个文件，导致此文件被两个进程同时监控。当其中一个进程按照配置滚动的时候，删除了最早的日志文件，但是另一个进程依旧占用此文件句柄。从而产生了 deleted 状态的文件。

处理步骤

将 spark 的 executor 日志输出目录修改成其他名称

1. 打开日志配置文件，默认在<客户端地址>/Spark/spark/conf/log4j-executor.properties。
2. 将日志输出文件改名，例如：

   log4j.appender.sparklog.File = ${spark.yarn.app.container.log.dir}/stdout 改为：
   log4j.appender.sparklog.File = ${spark.yarn.app.container.log.dir}/stdout.log
3. 保存退出
4. 重新提交任务。

2.8 如何解除关联子网

操作场景

您可根据自身网络需求，解除网络 ACL 与子网关联。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在系统首页，单击“网络 > 虚拟私有云”。

步骤 3 在左侧导航栏单击“网络 ACL”。

步骤 4 在右侧在“网络 ACL”列表区域，选择网络 ACL 的名称列，单击您需要修改的“网络 ACL 名称”进入网络 ACL 详情页面。

步骤 5 在详情页面，单击“关联子网”页签。

步骤 6 在“关联子网”页签详情区域，选择对应子网的“操作”列，单击“取消关联”。

步骤 7 单击“确认”。

----结束

2.9 修改 hostname，导致 MRS 状态异常

用户问题

修改 hostname 后，MRS 状态异常怎么处理？

问题现象

修改 hostname，导致 MRS 状态异常。

原因分析

修改 hostname 导致兼容性问题和故障。

处理步骤

步骤 1 以 root 用户登录集群的任意节点。

步骤 2 在集群节点中执行 **cat /etc/hosts** 命令，查看各个节点的 hostname 值，根据此值来配置 newhostname 变量值 。

步骤 3 在 hostname 被修改的节点上执行 **sudo hostnamectl set-hostname ${newhostname}**命令，恢复正确的 hostname。

说明

${newhostname}：表示新的 hostname 取值。

步骤 4 修改完成后，重新登录修改 hostname 的节点，查看修改的 hostname 是否生效。

----结束

2.10 如何定位进程被 kill

问题背景与现象

在某环境出现 DataNode 异常重启，且确认此时未从页面做重启 DataNode 的操作，需要定位是什么进程 kill DataNode 服务端进程。

原因分析

常见的进程被异常终止有 2 种原因：

- **Java 进程 OOM 被 Kill**

  一般 Java 进程都会配置 OOM Killer，当检测到 OOM 会自动 Kill，OOM 日志通常被打印到 out 日志中，此时可以看运行日志（如 DataNode 的日志路径为 /var/log/Bigdata/hdfs/dn/hadoop-omm-datanode-主机名.log），看是否有 OutOfMemory 内存溢出的打印。

- **被其他进程 kill，或者人为 kill。**

  排查 DataNode 运行日志（/var/log/Bigdata/hdfs/dn/hadoop-omm-datanode-主机名.log），是先收到“RECEIVED SIGNAL 15”再健康检查失败。即如下示例中 DataNode 先于 11:04:48 被 kill，然后过 2 分钟，于 11:06:52 启动。

```
2018-12-06 11:04: 48,433 | ERROR | SIGTERM handler | RECEIVED SIGNAL 15:
SIGTERM | LogAdapter.java:69
2018-12-06 11:04:48,436 | INFO  | Thread-1 | SHUTDOWN_MSG:
```

```
/************************************************************
SHUTDOWN_MSG: Shutting down DataNode at 192-168-235-85/192.168.235.85
************************************************************/ |
LogAdapter.java:45
2018-12-06 11:06:52,744 | INFO  | main | STARTUP_MSG:
```

以上日志说明，DataNode 先被其他进程关闭，然后健康检查失败，2 分钟后，被 NodeAgent 启动 DataNode 进程。

处理步骤

打开操作系统审计日志，给审计日志增加记录 kill 命令的规则，即可定位是何进程发送的 kill 命令。

操作影响

- 打印审计日志，会消耗一定操作系统性能，经过分析仅影响不到 1%。
- 打印审计日志，会占用一定磁盘空间。该日志打印量不大，MB 级别，且默认配置有老化机制和检测磁盘剩余空间机制，不会占满磁盘。

定位方法

在 DataNode 进程可能发生重启的所有节点，分别执行以下操作。

步骤 1 以 **root** 用户登录节点，执行 **service auditd status** 命令，确认该服务状态。

```
Checking for service auditd  running
```

如果该服务未启动，执行 **service auditd restart** 命令重启该服务（无影响，耗时不到 1 秒）

```
Shutting down auditd done
Starting auditd done
```

步骤 2 审计日志临时增加 kill 命令审计规则。

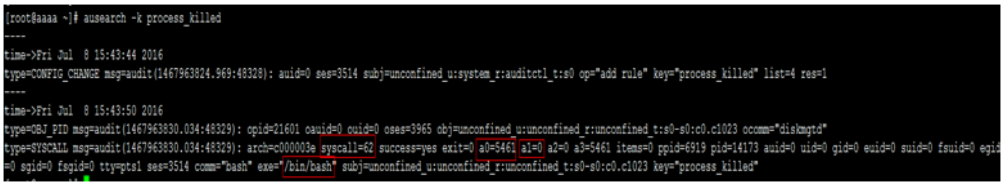
增加规则：

auditctl -a exit,always -F arch=b64 -S kill -S tkill -S tgkill -F a1!=0 -k process_killed

查看规则：

auditctl -l

步骤 3 当进程有异常被 kill 后，使用 **ausearch -k process_killed** 命令，可以查询 kill 历史。



说明

a0 是被 kill 进程的 pid（16 进制），a1 是 kill 命令的信号量。

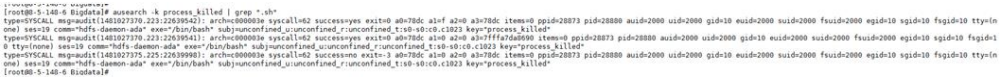
----结束

验证方法

步骤 1 从 MRS 页面重启该节点一个实例，如 DataNode。

步骤 2 执行 **ausearch -k process_killed** 命令，确认是否有日志打印。

例如以下命令 ausearch -k process_killed |grep “.sh” ，可以看到是 hdfs-daemon-ada* 脚本，关闭的 DataNode 进程。



----结束

停止审计 kill 命令方法

步骤 1 执行 **service auditd restart** 命令，即会清理临时增加的 kill 审计日志。

步骤 2 执行 **auditctl -l** 命令，如果没有 kill 相关信息，即说明已清理该规则。

----结束

2.11 MRS 集群使用 pip3 安装 python 包提示网络不可达

用户问题

使用 pip3 安装 python 包报错网络不可达。

问题现象

执行 pip3 install 安装 python 包报错网络不可达。具体如下图所示：

原因分析

客户未给 Master 节点绑定弹性公网 IP，造成报错的发生。

处理步骤

步骤 1 登录 MRS 服务管理控制台。

步骤 2 选择“集群列表 > 现有集群”，选中当前安装出问题的集群并单击集群名称，进入集群基本信息页面。

步骤 3 在“节点管理”页签单击 Master 节点组中某一 Master 节点名称，登录到弹性云主机管理控制台。

步骤 4 选择“弹性公网 IP”页签，单击“绑定弹性公网 IP”为弹性云主机绑定一个弹性公网 IP。

步骤 5 登录 Master 节点执行 pip3 install 安装 python 包。

----结束

2.12 开源 confluent-kafka-go 连接 MRS 的安全集群

用户问题

开源 confluent-kafka-go 如何连接 MRS 的安全集群？

问题现象

开源 confluent-kafka-go 连接 MRS 的安全集群失败。

原因分析

confluent-kafka-go 依赖的库 librdkafka 默认将 broker 所在 hostname 作为了 server princple 的一部分来使用，导致认证失败。

处理步骤

librdkafka 具体修改步骤：

1. librdkafka 源码地址：https://github.com/edenhill/librdkafka。
2. 在 src/rdkafka_conf.c 文件中增加 sasl.kerberos.service.name 配置项。

```
 "Kerberos principal name that Kafka runs as.",          .sdef = "kafka" },
{ _RK_GLOBAL, "sasl.kerberos.principal", _RK_C_STR,
_RK(sasl.principal),          "This client´s Kerberos principal
name.",          .sdef = "kafkaclient" }, +       { _RK_GLOBAL,
"sasl.kerberos.domain.name", _RK_C_STR, +         _RK(sasl.domain_name), +
"This cluster´s Kerberos domain name.", +         .sdef = "hadoop.hadoop.com" },
#ifndef _MSC_VER         { _RK_GLOBAL, "sasl.kerberos.kinit.cmd", _RK_C_STR,
_RK(sasl.kinit_cmd),          "Full kerberos kinit command
string, %{config.prop.name} is replaced "          "by corresponding config
object value, %{broker.name} returns the "          "broker´s hostname.", -
         .sdef = "kinit -S \"%{sasl.kerberos.service.name}/%{broker.name}\" "
+         .sdef = "kinit -S
\"%{sasl.kerberos.service.name}/%{sasl.kerberos.domain.name}\" "          "-k -
t \"%{sasl.kerberos.keytab}\" %{sasl.kerberos.principal}" },
{ _RK_GLOBAL, "sasl.kerberos.keytab", _RK_C_STR,          _RK(sasl.keytab),
"Path to Kerberos keytab file. Uses system default if not set."
"**NOTE**: This is not automatically used but must be added to the "
"template in sasl.kerberos.kinit.cmd as "
```

3. 在 src/rdkafka_conf.h 文件中增加 domain_name 字段。

```
--- src\rdkafka_conf.h       2017-10-17 11:20:56.000000000 +0800 +++
src\rdkafka_conf.h       2017-10-25 16:26:34.000000000 +0800 @@ -118,12 +118,13
@@         struct {                const struct rd_kafka_sasl_provider *provider;
```

```
char *principal;               char *mechanisms;               char
*service_name; +                          char *domain_name;               char
*kinit_cmd;              char *keytab;              int   relogin_min_time;
char *username;               char *password; #if WITH_SASL_SCRAM
```

4. 在 src/rdkafka_sasl_cyrus.c 文件中将 hostname 替换成 domainName。

```
--- src\rdkafka_sasl.c       2017-10-17 11:20:56.000000000 +0800 +++
src\rdkafka_sasl.c       2017-10-25 16:09:38.000000000 +0800 @@ -192,13 +192,14
@@                          rk->rk_conf.sasl.mechanisms,
rk->rk_conf.api_version_request ? "" :                          ": try
api.version.request=true");              return -1;         } -
rd_strdupa(&hostname, rktrans->rktrans_rkb->rkb_nodename); +
//rd_strdupa(&hostname, rktrans->rktrans_rkb->rkb_nodename); +
rd_strdupa(&hostname, rk->rk_conf.sasl.domain_name);         if ((t =
strchr(hostname, ':')))               *t = '\0';  /* remove ":port" */
```

5. 重新编译 librdkafka（请确保已安装 libsasl2-dev），具体步骤参考 https://github.com/edenhill/librdkafka/tree/v0.11.1。

   ./configure make make install

6. 使用客户端时增加如下配置项。

```
"security.protocol": "SASL_PLAINTEXT",
"sasl.kerberos.service.name": "kafka",
"sasl.kerberos.keytab": "/opt/nemon/user.keytab",
"sasl.kerberos.principal": "nemon@HADOOP.COM",
"sasl.kerberos.domain.name": "hadoop.hadoop.com",
```

说明

MRS 2.1.x 及之前版本：

- sasl.kerberos.keytab：可通过在 MRS Manager 界面选择“系统设置 > 用户管理”，在对应用户所在行的“操作”列选择“更多 >下载认证凭据”，保存后解压得到用户的 user.keytab 文件。
- sasl.kerberos.principal：请填写实际用户名。
- sasl.kerberos.domain.name：domain 的命名规则为 hadoop. toLowerCase(realm)，假设集群的域名（default_realm）为 xxx.com 时，domain 的值为 hadoop.xxx.com。可通过 MRS Manager 界面选择“服务管理> KrbServer > > 服务配置 > 全部配置” ，搜索并查看 default_realm 的值。

MRS 3.x 及后续版本：

- sasl.kerberos.keytab：可通过在 FusionInsight Manager 界面选择“系统 > 权限 > 用户”，在对应用户所在行的“操作”列选择“更多 >下载认证凭据”，保存后解压得到用户的 user.keytab 文件。
- sasl.kerberos.principal：请填写实际用户名。
- sasl.kerberos.domain.name：domain 的命名规则为 hadoop. toLowerCase(realm)，假设集群的域名（default_realm）为 xxx.com 时，domain 的值为 hadoop.xxx.com。可通过 FusionInsight Manager 界面选择“集群 > 服务 > KrbServer > > 配置 > 全部配置” ，搜索并查看 default_realm 的值。

2.13 MRS 集群客户端无法下载

用户问题

在本地的 Master 主机上想给另外一台远端主机下载一个 MRS 集群客户端进行使用，但是一直提示网络或者参数有问题

问题现象

在本地的 Master 主机上想给另外一台远端主机下载一个 MRS 集群客户端进行使用，但是一直提示网络或者参数有问题

原因分析

- 可能是两台主机处于不同 VPC 网络中
- 密码填写错误
- 远端主机开启防火墙

处理步骤

- 两台主机处于不同 VPC 网络中

  放开远端主机的 22 端口
- 密码填写错误

  请检查密码是否正确，密码中不能有特殊符号。
- 远端主机开启防火墙

  使用规避方案，先将这个 MRS 集群客户端下载到“服务器端”主机，然后通过 linux 提供的 scp 命令远程发送到远端主机。

2.14 开启 Kerberos 认证集群提交 Flink 作业报错

用户问题

用户开启 Kerberos 认证提交 flink 作业报错。

问题现象

客户提交 flink 官方案例./flink run /opt/client/Flink/flink/examples/streaming/WordCount.jar 报错：unable to establish the security context。

原因分析

1. 客户开启了 kerberos 认证但是无法提交作业，所以首先检查权限配置问题，检查发现未正确配置/opt/client/Flink/flink/conf/flink-conf.yaml”中的参数。

图2-8 flink-conf.yaml 配置

```
query.proxy.query-threads: 0
security.kerberos.login.keytab: /data01/fiClient/mrs_210_new/keytab/flinkuser/user.keytab
high-availability.zookeeper.client.retry-wait: 5000
taskmanager.runtime.hashjoin-bloom-filters: false
akka.lookup.timeout: 10 s
blob.fetch.backlog: 1000
fs.obs.socket.send.buffer: 65536
blob.service.ssl.enabled: true
blob.fetch.num-concurrent: 50
query.server.query-threads: 0
akka.watch.heartbeat.interval: 10 s
security.kerberos.login.use-ticket-cache: false
fs.obs.max.total.tasks: 20
fs.obs.fast.upload: true
yarn.application-attempts: 3
akka.ask.timeout: 10 s
taskmanager.debug.memory.log-interval: 0
fs.obs.buffer.max.range: 136000
fs.obs.threads.read.core: 500
security.ssl.protocol: TLSv1.2
restart-strategy.fixed-delay.attempts: 3
akka.throughput: 15
taskmanager.network.netty.server.numThreads: 3
taskmanager.rpc.port: 32326-32390
akka.startup-timeout: 60 s
task.checkpoint.alignment.max-size: -1
taskmanager.runtime.max-fan: 128
web.ssl.enabled: false
containerized.heap-cutoff-ratio: 0.25
taskmanager.runtime.sort-spilling-threshold: 0.8
security.ssl.internal.keystore: /mrs_210_new/Flink/flink/conf//flink.keystore
fs.obs.endpoint: obs.cn-north-5.myhuaweicloud.com
```

keytab文件路径

此处配置为相对路径，若为绝对路径每个节点都要配置。

2. 修改并刷新配置后，重新提交作业出现作业可以提交但报 log4j:ERROR setFile(null,true) call failed 的错误。

图2-9 log4j 报错

```
        at org.apache.flink.runtime.entrypoint.ClusterEntrypoint.<clinit>(ClusterEntrypoint.java:98)
log4j:ERROR setFile(null,true) call failed.
```

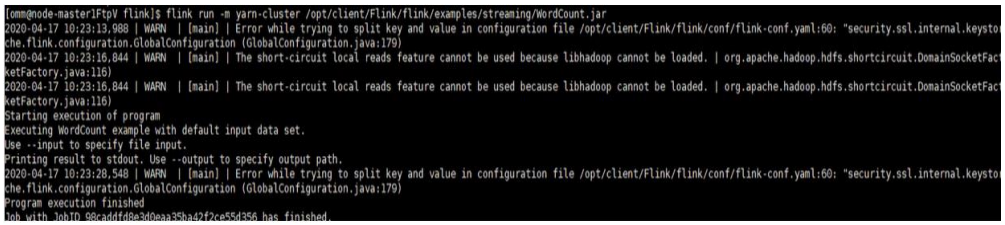
3. 发现是 log4j 相关的报错，查看 log4j 发现客户将 log4j.properties 文件改成了 log4g-cli.properties（log4j.properties 的名字是固定的不可随意修改）于是报错。

图2-10 查看 log4j

```
[emr@kwetemr00005 flink]$ cd conf/
[emr@kwetemr00005 conf]$ ll
total 52
-rwxr-xr-x 1 emr emr  727 Apr 15 10:26 ca.cer
-rwxr-xr-x 1 emr emr 2075 Apr 15 10:26 ca.keystore
-rwxr-xr-x 1 emr emr  358 Apr 15 08:43 client.properties
-rwxr-xr-x 1 emr emr  534 Apr 15 08:43 core-site.xml
-rwxr-xr-x 1 emr emr  773 Apr 15 10:26 flink.cer
-rwxr-xr-x 1 emr emr 5564 Apr 20 10:12 flink-conf.yaml
-rwxr-xr-x 1 emr emr  965 Apr 15 10:26 flink.csr
-rwxr-xr-x 1 emr emr 1739 Apr 15 08:43 hdfs-site.xml
-rwxr-xr-x 1 emr emr 1113 Apr 15 10:47 log4j-cli.properties
-rwxr-xr-x 1 emr emr  976 Apr 15 08:43 log4j.properties
-rwxr-xr-x 1 emr emr  709 Apr 15 08:43 log4j-yarn-session.properties
drwxrwxr-x 2 emr emr   50 Apr 17 09:50 ssl
-rwxr-xr-x 1 emr emr 1953 Apr 15 08:43 yarn-site.xml
```

4. 修改后可以正常提交作业。

图2-11 提交作业正常



处理步骤

步骤 1 判断用户是在集群外还是集群内使用客户端提交作业。

1. 若在集群内使用客户端，切换到 omm 用户提交作业。
2. 若在集群外使用客户端，则要使用 root 用户提交作业。

步骤 2 检查 flink-conf.yaml 各参数是否配置正确。

步骤 3 对于开启 Kerberos 认证的集群配置项包括 Kerberos 的 keytab、principal 等。

- 从 KDC 服务器上下载用户 keytab，并将 keytab 放到 Flink 客户端所在主机的某个文件夹下(例如/home/flinkuser/keytab)。
- 在“${FLINK_HOME}/conf/flink-conf.yaml”上配置：
  a. keytab 路径（注意配置参数前面有空格）：

  ```
  security.kerberos.login.keytab:  /home/flinkuser/keytab/uer.keytab
  ```

  b. principal 名(即开发用户名)：

  ```
  security.kerberos.login.principal:flinkuser
  ```

步骤 4 重新正确提交作业**./flink run /opt/client/Flink/flink/examples/streaming/WordCount.jar**，验证是否可以提交作业。

- 若可以提交作业则说明权限认证没有问题，就可以去检查其他错误，本例中是修改了 log4j.properties 的名称，还原后可以正常提交作业。
- 若提交作业失败，请。

----结束

参考信息

Flink 使用请参考。

2.15 扩容失败

用户问题

Console 界面正常，MRS 集群扩容失败

问题现象

Console 界面正常，查看 MRS Manager 界面也无警告无错误，但 MRS 集群无法扩容报“集群存在非运行状态节点，请稍后重试”的错误。

原因分析

MRS 集群的扩缩容要建立在集群处于正常运行的基础上，所以首先要检查集群是否处于正常与否，现在报的是集群存在非运行状态节点，而 console 界面和 MRS Manager 界面都是正常的，所以可能原因就是数据库中集群状态不正常或未刷新导致集群相关节点处于非正常运行状态导致的。

处理步骤

步骤 1 登录 MRS 控制台，单击集群名称进入集群详情页面查看集群状态，确保集群状态为“运行中”。

步骤 2 单击“节点管理”，查看所有节点的状态，确保所有节点的状态为“运行中”。

步骤 3 登录集群的 podMaster 节点跳转到 MRS 的 deployer 节点，查看 api-gateway.log 的日志。

1. 用 **kubectl get pod -n mrs** 命令查看 MRS 对应的 **deployer** 节点的 **pod**。
2. 用 **kubectl exec -ti ${deployer 节点的 pod} -n mrs /bin/bash** 命令登录相应的 pod，如执行 **kubectl exec -ti mrsdeployer-78bc8c76cf-mn9ss -n mrs /bin/bash** 命令进入 MRS 的 deployer 容器。
3. 在/opt/cloud/logs/apigateway 目录下查看最新的 api-gateway.log 日志，检索里面的关键信息（如：ERROR，scaling，clusterScaling，HostState，state-check，集群 ID 等)查看报错类型。
4. 根据报错提示信息进行相应处理，然后再次执行扩容操作。
   - 扩容成功，则处理完成。
   - 扩容失败，则执行步骤 4。

步骤 4 用**/opt/cloud/mysql -u${用户名} -P${端口} -h${地址} -p${密码}**登录数据库。

步骤 5 执行 **select cluster_state from cluster_detail where cluster_id="集群 ID";**查看 cluster_state。

- cluster_state 为 2，则集群状态正常，执行步骤 6。
- cluster_state 不为 2，说明集群状态在数据库中异常，可用 **update cluster_detail set cluster_state=2 where cluster_id="集群 ID";**刷新集群状态，并查看 cluster_state。
  - cluster_state 为 2，则集群状态正常，执行步骤 6
  - cluster_state 不为 2，则请。

步骤 6 执行 **select host_status from host where cluster_di="clusterID";**命令查询集群主机状态。

- 如果主机状态为 started，则处理完成。
- 如果主机状态不为 started，则可执行 **update host set host_status='started' where cluster_id="集群 ID";**命令更新主机状态到数据库。
  - 如果主机状态为 started，则处理完成。

- 如果主机状态不为 started，则请。

----结束

2.16 MRS 通过 beeline 执行插入命令的时候出错

用户问题

MRS 通过 beeline 执行插入命令的时出错

问题现象

在 hive 的 beeline 中执行 **insert into** 插入语句的时候会报以下的错误:

```
Mapping run in Tez on Hive transactional table fails when data volume is high with
error: "org.apache.hadoop.hive.ql.lockmgr.LockException Reason: Transaction...
already aborted, Hive SQL state [42000]."
```

原因分析

对于 Join 操作，由于集群配置不理想和 Tez 资源设置不合理导致该问题。

处理步骤

可以在 beeline 上设置配置参数进行解决。

步骤 1 设置以下属性以优化性能（建议在集群级别进行更改）

- 设置 hive.auto.convert.sortmerge.join = true
- 设置 hive.optimize.bucketmapjoin = true
- 设置 hive.optimize.bucketmapjoin.sortedmerge = true

步骤 2 更改以下内容以调整 Tez 的资源 。

- 设置 hive.tez.container.size = {与 YARN 容器相同的大小}
- 将 hive.tez.container.size 设置为与 YARN 容器大小 yarn.scheduler.minimum-allocation-mb 相同或更小的值（如设置为二分之一或四分之一的值），但不要超过 yarn.scheduler.maximum-allocation-mb 。

----结束

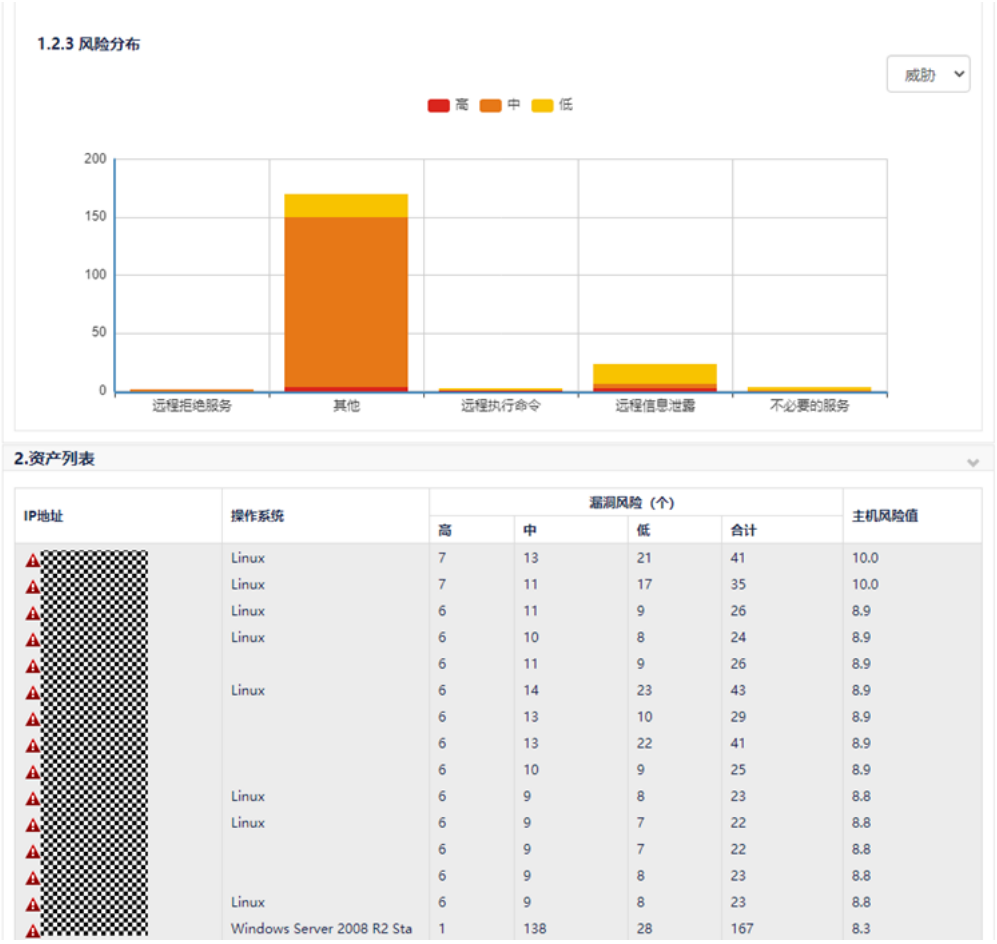
2.17 MRS 集群如何进行 Euleros 系统漏洞升级？

用户问题

Euleros 系统底层存在漏洞，MRS 集群如何进行漏洞升级？

问题现象

在使用绿盟软件测试集群，发现有 Euleros 系统底层存在漏洞，漏洞报告如下：



原因分析

在使用绿盟软件测试集群，发现有 Euleros 系统底层存在漏洞，MRS 服务部署在 Euleros 系统中，因此需要进行漏洞升级。

处理步骤

说明

修复漏洞前请确认是否开启了企业主机安全（Host Security Service，简称 HSS）服务，如果已开启，需要先暂时关闭 HSS 服务对 MRS 集群的监测，漏洞修复完成后重新开启 HSS 服务。

步骤 1　登录 MRS 服务控制台。

步骤 2　单击集群名称进入集群详情页面，并选择“节点管理”。

步骤 3　在 Core 节点组中勾选任意一个 Core 节点，单击“节点操作 > 停止所有角色”。

步骤 4 通过远登录 Core 节点后台，配置 yum 源。

步骤 5 使用 **uname －r** 或 **rpm －qa |grep kernel** 命令，查询并记录当前节点内核版本。

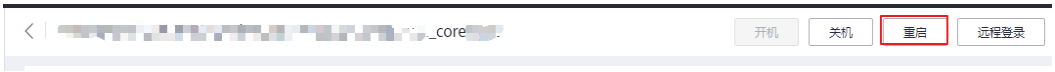
步骤 6 执行 **yum update －y --skip-broken --setopt=protected_multilib=false** 命令更新补丁。

步骤 7 完成更新后查询内核版本，并执行 **rpm -e** *旧内核版本* 命令删除旧内核版本。

步骤 8 在集群详情页，选择“节点管理”。

步骤 9 在 Core 节点组中单击已更新补丁的 Core 名称，进入弹性云服务管理控制台。

步骤 10 在页面右上角单击“重启”，重启 Core 节点。



步骤 11 重启完成后，在集群详情页的“节点管理”的 Core 节点组中勾选 Core 节点，单击“节点操作 > 启动所有角色”。

步骤 12 重复步骤 1~步骤 11 的操作，升级其他 Core 节点。

步骤 13 所有 Core 节点升级完成后，参考步骤 1~步骤 11 的操作先升级备 Master 节点，再升级主 Master 节点。

----结束

2.18 使用 CDM 迁移数据至 HDFS

用户问题

使用 CDM 从旧的集群迁移数据至新集群的 HDFS 过程失败。

问题现象

使用 CDM 从源 HDFS 导入目的端 HDFS，发现目的端 MRS 集群故障，NameNode 无法启动。

查看日志发现在启动过程中存在 Java heap space 报错，需要修改 NN 的 JVM 参数。

图2-12 故障日志

原因分析

客户在使用 CDM 迁移数据的过程中，HDFS 的数据量过大，导致在合并元数据时发生堆栈异常。

处理步骤

步骤 1 搜索并修改“HDFS->NameNode” 中的“GC_OPTS”参数，将其中的“-Xms512M -Xmx512M”两个参数的值根据实际情况调整为较大的值。

步骤 2 保存配置并重启受影响的服务或实例。

----结束

2.19 MRS 集群频繁产生告警

用户问题

集群频繁发出 Manager 主备节点间心跳中断，DBService 主备节点间心跳中断，节点故障等告警，偶尔会造成 hive 不可用。

问题现象

集群频繁发出 Manager 主备节点间心跳中断，DBService 主备节点间心跳中断，节点故障等告警，偶尔会造成 hive 不可用，影响客户业务。

原因分析

1. 在出现告警时间点发现虚拟机发生了重启，告警发生的原因是因虚拟机重启导致的。

```
[omm@node-master1yqIY nodeagent]$ last
omm      pts/0        100.125.0.70     Thu Sep 24 10:33   still logged in
omm      pts/1        100.125.0.70     Thu Sep 24 09:26 - 09:47  (00:20)
omm      pts/0        100.125.0.70     Thu Sep 24 09:22 - 10:21  (00:59)
omm      pts/1        100.125.0.70     Wed Sep 23 17:32 - 17:37  (00:05)
root     pts/0        10.203.216.102   Wed Sep 23 17:13 - 18:35  (01:21)
omm      pts/0        100.125.0.70     Wed Sep 23 16:55 - 16:56  (00:00)
omm      pts/0        100.125.0.70     Wed Sep 23 16:20 - 16:25  (00:05)
reboot   system boot  4.19.36-vhulk190 Wed Sep 23 16:06   still running
root     pts/1        10.203.216.102   Tue Sep 22 19:13 - 19:48  (00:34)
omm      pts/0        100.125.0.70     Tue Sep 22 19:08 - 20:03  (00:54)
root     pts/0        10.203.216.102   Tue Sep 22 17:03 - 17:52  (00:48)
omm      pts/1        100.125.0.70     Tue Sep 22 15:55 - 16:00  (00:05)
```

```
[omm@node-master2WbYp ~]$ last
omm      pts/0        10.80.0.56       Thu Sep 24 11:00   still logged in
omm      pts/0        10.80.0.56       Thu Sep 24 09:24 - 10:21  (00:56)
omm      pts/0        10.80.0.56       Wed Sep 23 17:32 - 17:37  (00:05)
omm      pts/0        10.80.0.56       Tue Sep 22 19:15 - 19:15  (00:00)
omm      pts/0        10.80.0.56       Tue Sep 22 15:57 - 16:21  (00:23)
omm      pts/0        10.80.0.56       Tue Sep 22 15:23 - 15:35  (00:12)
omm      pts/0        10.80.0.56       Tue Sep 22 15:07 - 15:12  (00:05)
omm      pts/0        10.80.0.56       Tue Sep 22 14:21 - 14:26  (00:05)
omm      pts/0        10.80.0.56       Mon Sep 21 10:57 - 11:06  (00:09)
omm      pts/0        10.80.0.56       Mon Sep 21 10:42 - 10:56  (00:14)
omm      pts/0        10.80.0.56       Thu Sep 17 16:05 - 16:15  (00:10)
omm      pts/0        10.80.0.56       Wed Sep 16 20:52 - 20:58  (00:06)
reboot   system boot  4.19.36-vhulk190 Wed Sep 16 18:05   still running
omm      pts/0        10.80.0.56       Wed Sep 16 15:43 - 16:10  (00:26)
omm      pts/0        10.80.0.56       Wed Sep 16 14:35 - 14:53  (00:17)
omm      pts/0        10.80.0.56       Wed Sep 16 14:33 - 14:33  (00:00)
omm      pts/0        10.80.0.56       Wed Sep 16 14:11 - 14:29  (00:17)
omm      pts/0        10.80.0.56       Wed Sep 16 14:02 - 14:09  (00:06)
omm      pts/0        10.80.0.56       Wed Sep 16 11:56 - 12:04  (00:08)
omm      pts/0        10.80.0.56       Wed Sep 16 11:26 - 11:31  (00:04)
omm      pts/0        10.80.0.56       Wed Sep 16 11:09 - 11:24  (00:15)
root     pts/0        10.203.230.193   Mon Sep 14 15:54 - 16:30  (00:35)
root     pts/0        10.203.172.29    Fri Sep 11 17:15 - 17:45  (00:30)
root     pts/0        10.203.172.29    Fri Sep 11 16:53 - 17:12  (00:19)
root     tty1                          Fri Sep 11 16:23 - 17:25  (01:01)
reboot   system boot  4.19.36-vhulk190 Fri Sep 11 10:07   still running
reboot   system boot  4.19.36-vhulk190 Thu Aug 27 16:41   still running
root     tty1                          Thu Aug 20 09:46 - 10:17  (00:30)
reboot   system boot  4.19.36-vhulk190 Wed Aug 19 17:48   still running
reboot   system boot  4.19.36-vhulk190 Wed Aug 19 17:46   still running
```

2. 经 OS 定位虚拟机发生重启的原因是节点没有可用的内存，系统发生内存溢出触发了 oom-killer，当进程处于被调用的状态会使进程处于 disk sleep 状态，最终导致虚拟机发生重启。

```
mem info:
[344766.903734] MemTotal:       32397404 kB
MemFree:          160404 kB
MemAvailable:      31668 kB
Buffers:            2172 kB
Cached:          2768904 kB
SwapCached:            0 kB
Active:         30328872 kB
Inactive:        1035844 kB
Active(anon):   30320852 kB
Inactive(anon):  1004376 kB
Active(file):       8020 kB
Inactive(file):    31468 kB
Unevictable:           0 kB
Mlocked:               0 kB
[344766.903738] SwapTotal:             0 kB
SwapFree:              0 kB
```

总内存

用户态使用

3. 查看占用的内存进程，发现占用内存都是正常的业务进程。

结论：虚拟机内存不能满足服务需求。

处理步骤

- 建议扩大节点内存。
- 建议关闭不需要的服务来规避该问题。

2.20 PMS 进程占用内存高问题处理

用户问题

主 Master 节点内存使用率高如何处理？

问题现象

主 Master 节点内存使用率高，且用 **top -c** 命令查询得内存占用量高的是如下 idle 的进程。

```
12180 ommdba    20   0 1395492 1.180g 1.082g S   0.0  3.8  23:14.29 gaussdb: OMM OMM localhost(60598) idle
14828 ommdba    20   0 1395904 1.180g 1.081g S   0.0  3.8  23:17.08 gaussdb: OMM OMM localhost(60698) idle
15016 ommdba    20   0 1395840 1.180g 1.081g S   0.0  3.8  23:11.19 gaussdb: OMM OMM localhost(60824) idle
14943 ommdba    20   0 1395900 1.180g 1.081g S   0.0  3.8  23:14.76 gaussdb: OMM OMM localhost(60764) idle
14908 ommdba    20   0 1395840 1.180g 1.081g S   0.0  3.8  23:15.18 gaussdb: OMM OMM localhost(60738) idle
14953 ommdba    20   0 1395824 1.180g 1.081g S   0.0  3.8  23:15.96 gaussdb: OMM OMM localhost(60770) idle
14995 ommdba    20   0 1395560 1.180g 1.081g S   0.0  3.8  23:13.28 gaussdb: OMM OMM localhost(60812) idle
15062 ommdba    20   0 1395820 1.180g 1.081g S   0.0  3.8  23:16.12 gaussdb: OMM OMM localhost(60868) idle
15064 ommdba    20   0 1395512 1.180g 1.081g S   0.0  3.8  23:13.33 gaussdb: OMM OMM localhost(60870) idle
14973 ommdba    20   0 1395528 1.180g 1.081g S   0.0  3.8  23:12.74 gaussdb: OMM OMM localhost(60790) idle
14835 ommdba    20   0 1395536 1.180g 1.081g S   0.0  3.8  23:17.39 gaussdb: OMM OMM localhost(60704) idle
14822 ommdba    20   0 1395524 1.180g 1.081g S   0.0  3.8  23:13.80 gaussdb: OMM OMM localhost(60692) idle
14991 ommdba    20   0 1395808 1.180g 1.081g S   0.0  3.8  23:17.96 gaussdb: OMM OMM localhost(60808) idle
14975 ommdba    20   0 1395812 1.180g 1.081g S   0.0  3.8  23:12.57 gaussdb: OMM OMM localhost(60792) idle
15038 ommdba    20   0 1395520 1.180g 1.081g S   0.0  3.8  23:12.75 gaussdb: OMM OMM localhost(60846) idle
14919 ommdba    20   0 1395540 1.180g 1.081g S   0.0  3.8  23:11.58 gaussdb: OMM OMM localhost(60744) idle
14832 ommdba    20   0 1395476 1.180g 1.081g S   0.0  3.8  23:13.11 gaussdb: OMM OMM localhost(60702) idle
14989 ommdba    20   0 1395500 1.180g 1.081g S   0.0  3.8  23:15.63 gaussdb: OMM OMM localhost(60806) idle
14979 ommdba    20   0 1395448 1.180g 1.081g S   0.0  3.8  23:13.17 gaussdb: OMM OMM localhost(60796) idle
15047 ommdba    20   0 1395512 1.180g 1.081g S   0.0  3.8  23:12.10 gaussdb: OMM OMM localhost(60854) idle
14977 ommdba    20   0 1395496 1.180g 1.081g S   0.0  3.8  23:16.90 gaussdb: OMM OMM localhost(60794) idle
15028 ommdba    20   0 1395800 1.180g 1.081g S   0.0  3.8  23:09.35 gaussdb: OMM OMM localhost(60836) idle
```

原因分析

- PostgreSQL 缓存：除了常见的执行计划缓存、数据缓存，PostgreSQL 为了提高生成执行计划的效率，还提供了 catalog，relation 等缓存机制。长连接场景下这些缓存中的某些缓存是不会主动释放的，因此可能导致长连接占用大量的内存不释放。
- PMS 是 MRS 的监控进程，此进程会经常创建表分区或者新表，由于 PostgreSQL 会缓存当前会话访问过的对象的元数据，且 PMS 的数据库连接池连接会长时间存在，所以连接占用的内存会逐渐上升。

处理步骤

步骤 1 以 **root** 用户登录主 Master 节点。

步骤 2 执行如下命令查询 PMS 进程号。

ps -ef | grep =pmsd |grep -v grep

步骤 3 执行如下命令关闭 PMS 进程，其中 **PID** 为步骤 2 中获取的 PMS 进程号。

kill -9 PID

步骤 4 等待 PMS 进程自动启动。

PMS 启动需要 2-3 分钟。PMS 是监控进程，重启不影响大数据业务。

----结束

2.21 Knox 进程占用内存高

用户问题

knox 进程占用内存高

问题现象

主 Master 节点内存使用率高，用 **top -c** 命令查看到占用内存较高的进程中有 knox 进程，且此进程占用内存超过 4G。

原因分析

knox 进程没有单独配置内存，进程会自动根据系统内存大小按照比例划分可用内存，导致 knox 占用内存大。

处理步骤

步骤 1 以 root 用户分别登录 Master 节点。

步骤 2 打开文件“/opt/knox/bin/gateway.sh”，查找 APP_MEM_OPTS，并设置该参数的值为：“-Xms3072m -Xmx4096m”。

步骤 3 登录 Manager 页面，单击“主机管理”，找到主 Master 节点的 IP（即主机名称前带有实心五角星的节点），并登录该节点后台。

步骤 4 执行如下命令重启进程。

su - omm

sh /opt/knox/bin/restart-knox.sh

----结束

2.22 安全集群外节点安装客户端访问 HBase 很慢

用户问题

安全集群外节点安装了集群的客户端，并使用客户端命令 hbase shell 访问 hbase，发现访问 hbase 非常慢。

问题现象

客户创建了安全集群，在集群外节点安装了集群的客户端，并使用客户端命令 hbase shell 访问 hbase，发现访问 hbase 非常慢。

原因分析

安全集群需要进行 Kerberos 认证，需要在客户端节点的 hosts 中配置信息，访问速度才不会收到影响。例如，hosts 配置信息为：

```
1.1.1.1 hadoop.782670e3_1364_47e2_8c70_1b61bb80479c.com
1.1.1.1 hadoop.hadoop.com
1.1.1.1 hacluster
1.1.1.1 haclusterX
1.1.1.1 haclusterX1
1.1.1.1 haclusterX2
1.1.1.1 haclusterX3
1.1.1.1 haclusterX4
1.1.1.1 ClusterX
1.1.1.1 manager
ip1 hostname1
ip2 hostname2
```

```
ip3 hostname3
ip4 hostname4
```

处理步骤

将集群节点上的 hosts 文件内容复制到安装客户端节点的 hosts 文件中。

2.23 作业无法提交如何定位？

问题背景与现象

客户通过 DataArts Studio 或者在 MRS Console 无法提交作业。

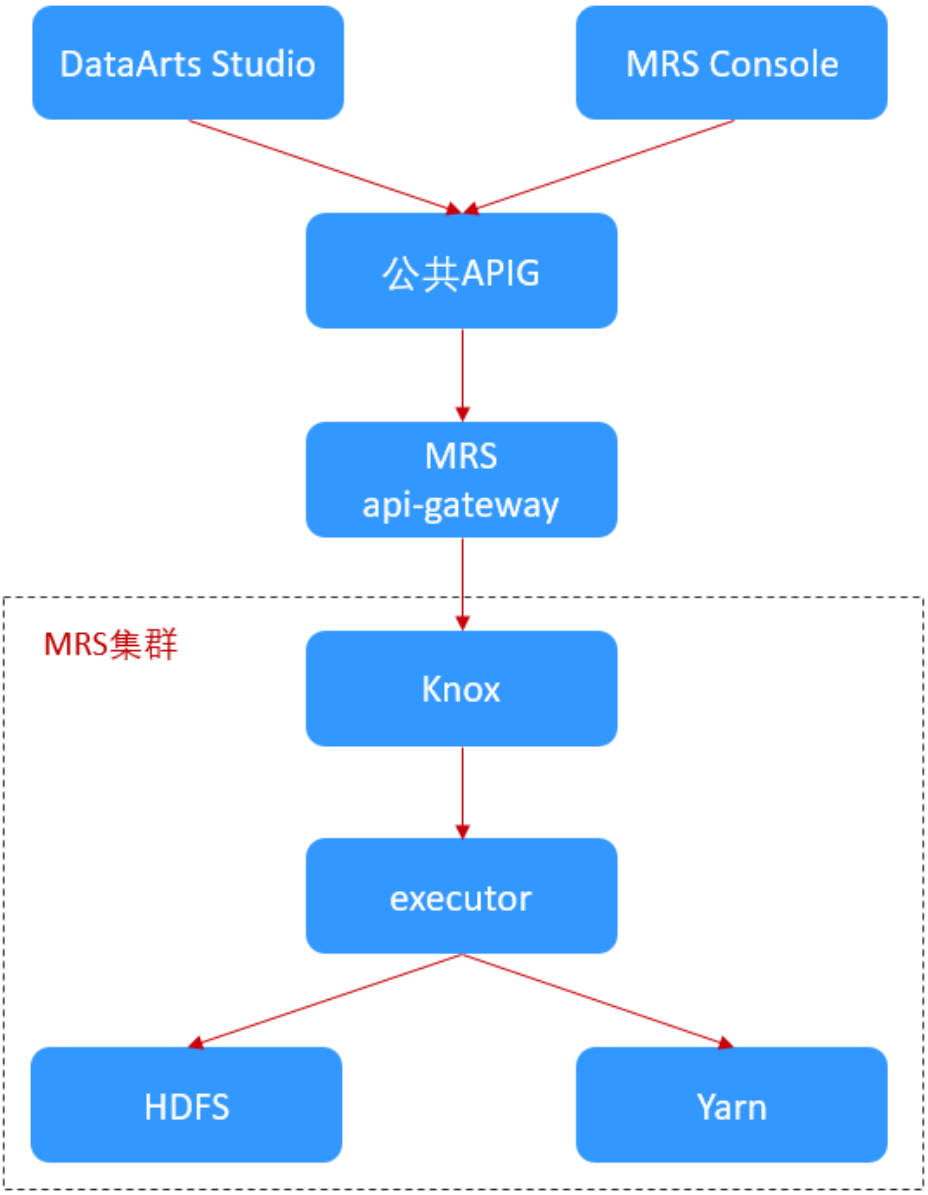
问题影响

作业无法提交，业务中断。

作业流程简介

1. 所有请求会先经过 APIG 网关，受到 APIG 配置的流控限制。
2. APIG 将请求转发到 MRS 管控面的 api-gateway 中。
3. MRS 管控面 API 节点轮询主备 oms 的 Knox，确认主 oms 的 Knox。
4. MRS 管控面 API 提交任务到主 oms 的 Knox。
5. Knox 转发请求到本节点的 executor 进程。
6. executor 进程提交任务到 Yarn。

图2-13 作业流程



处理步骤

前期准备：

- 确定作业是通过 DataArts Studio 或在 MRS Console 提交。
- 准备如表 2-1 信息。

表2-1 修复前准备事项

序号	项目	操作方式
1	集群帐号信息	申请集群 **admin** 账户的密码。

序号	项目	操作方式
2	节点帐号信息	申请集群内节点的 **omm**、**root** 用户密码。
3	SSH 远程登录工具	准备 PuTTY 或 SecureCRT 等工具。
4	客户端	已提前安装好客户端。

步骤 1 确认异常来源。

查看作业日志中收到的错误码，确认错误码是属于 APIG 还是 MRS。

- 若是公共 APIG 的错误码（APIG 的错误码是 APIGW 开头），联系公共 APIG 维护人员。
- 若是 MRS 侧错误，继续下一步。

步骤 2 排查服务和进程运行状态等基本情况。

1. 登录 Manager 界面确认是否有服务故障，如果有作业相关服务故障或者底层基础服务故障，需要解决故障。
2. 查看是否有严重告警。
3. 登录主 Master 节点。
4. 执行如下命令查看 oms 状态是否正常，主 oms 节点 executor 和 knox 进程是否正常。knox 是双主模式，executor 是单主模式。

   /opt/Bigdata/om-0.0.1/sbin/status-oms.sh
5. 以 omm 用户执行 **jmap -heap PID** 检查 knox 和 executor 进程内存使用情况，如果多次执行查看到老生代内存使用率为 99.9%说明有内存溢出。

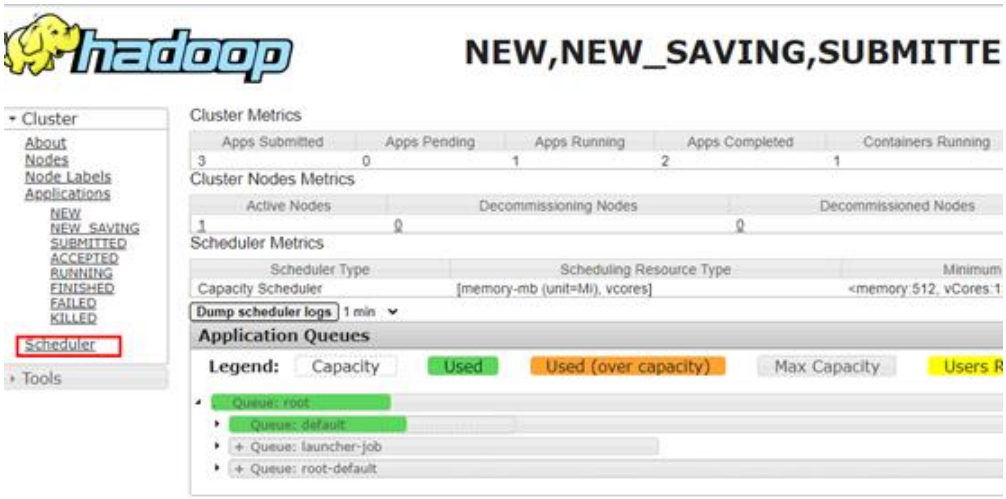
   查询 executor 进程 PID：netstat -anp | grep 8181 | grep LISTEN

   查询 knox 进程 PID：ps -ef|grep knox | grep -v grep

   如果内存溢出，需要现在执行 **jmap -dump:format=b,file=/home/omm/temp.bin PID**，导出内存信息后重启进程进行恢复。
6. 查看 Yarn 的原生界面，确认队列资源情况，以及任务是否提交到了 yarn 上。

   Yarn 的原生界面：在集群详情页选择“组件管理 > Yarn > ResourceManager WebUI > ResourceManager (主)”

图2-14 Yarn 界面队列资源情况



步骤 3 排查任务提交失败点。

1. 登录 MRS 控制台，单击集群名称进入集群详情页面。
2. 选择“作业管理”页签，单击作业所在行“操作”列的“查看日志”。
3. 若没有日志或者日志信息不详细，则在“作业名称/ID”列复制作业 ID。
4. 在主 oms 节点执行如下命令确认任务请求是否下发到了 knox，如果请求没有到 knox 则可能是 knox 出了问题，需要尝试重启 knox 进行恢复。

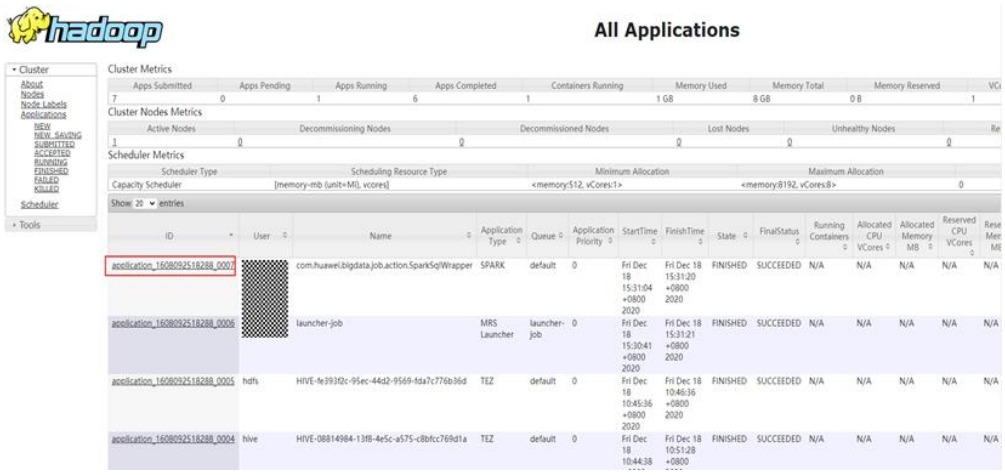
   grep "mrsjob" /var/log/Bigdata/knox/logs/gateway-audit.log | tail -10
5. 进入 executor 的日志中搜索作业 ID，查看报错信息。

   日志路径：/var/log/Bigdata/executor/logs/exe.log
6. 修改“/opt/executor/webapps/executor/WEB-INF/classes/log4j.properties”文件开启 executor 的 debug 日志，提交测试任务，查看 executor 的日志并确认作业提交过程中的报错。

   日志路径：/var/log/Bigdata/executor/logs/exe.log
7. 如果当前任务在 exeutor 中出错，执行如下命令打印 executor 的 jstack 信息，确认线程当前执行状态。

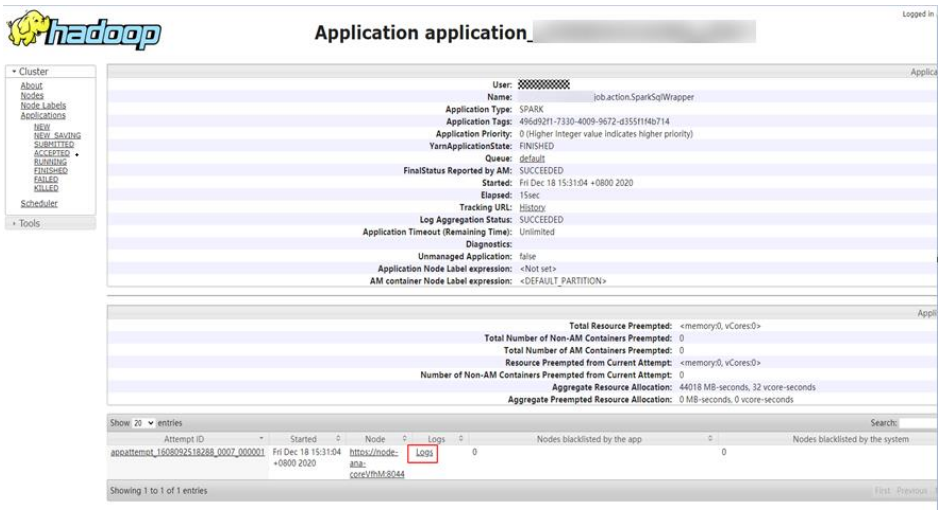
   jstack PID > xxx.log
8. 在集群详情页面选择“作业管理”页签，单击作业所在行“操作”列的“查看详情”，获取“实际作业编号”applicationID。
9. 在集群详情页选择“组件管理 > Yarn > ResourceManager WebUI > ResourceManager (主)”进去 Yarn 的原生界面，单击 applicationID。

图2-15 Yarn 的 Applications



10. 在任务详情页面查看日志。

图2-16 任务日志



----结束

2.24 HBase 日志文件过大导致 OS 盘空间不足

用户问题

OS 盘/var/log 分区空间不足。

问题现象

“/var/log/Bigdata/hbase/*/hbase-omm-*.out”日志文件过大，造成 OS 盘/var/log 分区空间不足。

原因分析

在 HBase 长时间运行场景下，操作系统会把 JVM 创建的“/tmp/.java_pid*”文件定期清理。因为 HBase 的内存监控使用了 JVM 的 jinfo 命令，而 jinfo 依赖“/tmp/.java_pid*”文件，当该文件不存在时，jinfo 会执行 **kill -3** 将 jstack 信息打印到.out 日志文件里，从而导致.out 日志文件过大。

处理步骤

在每个 HBase 实例的节点上部署定期清理.out 日志文件的定时任务。后台登录 HBase 的实例节点，在 crontab -e 中添加每天 0 点清理.out 日志的定时任务。

crontab -e

00 00 * * * for file in `ls /var/log/Bigdata/hbase/*/hbase-omm-*.out`; do echo "" > $file; done

说明

如果.out 大文件出现比较频繁，可以每天清理多次或者调整操作系统的自动清理策略。

2.25 HDFS 日志文件过大导致 OS 盘空间不足

用户问题

OS 盘/var/log 分区空间不足。

问题现象

“/var/log/Bigdata/hdfs/*/hdfs-omm-*.out”日志文件过大，造成 OS 盘/var/log 分区空间不足。

原因分析

在 HDFS 长时间运行场景下，操作系统会把 JVM 创建的“/tmp/.java_pid*”文件定期清理。因为 HDFS 的内存监控使用了 JVM 的 jinfo 命令，而 jinfo 依赖“/tmp/.java_pid*”文件，当该文件不存在时，jinfo 会执行 **kill -3** 将 jstack 信息打印到.out 日志文件里，从而导致.out 日志文件过大。

处理步骤

在每个 HDFS 实例的节点上部署定期清理.out 日志文件的定时任务。后台登录 HDFS 的实例节点，在 crontab -e 中添加每天 0 点清理.out 日志的定时任务。

crontab -e

00 00 * * * for file in `ls /var/log/Bigdata/hdfs/*/hdfs-omm-*.out`; do echo "" > $file; done

说明

如果.out 大文件出现比较频繁，可以每天清理多次或者调整操作系统的自动清理策略。

2.26 MRS 集群 Master 节点规格升级异常

用户问题

升级 Master 节点出现异常如何处理。

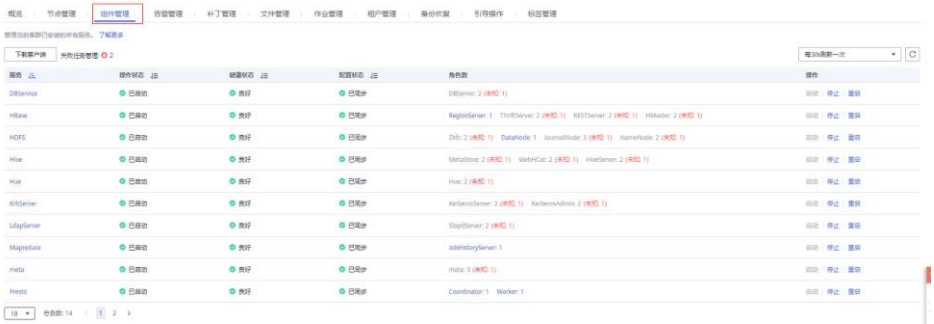
问题现象

升级 Master 节点可能存在规格升级成功后组件启动失败或组件异常导致节点升级后出现异常的情况，现象如下：

- 升级失败的节点规格已经升级成功。

- 失败任务管理中有新增的升级规格失败任务。

- 若 IAM 用户已同步，可在组件管理观察到存在异常角色。



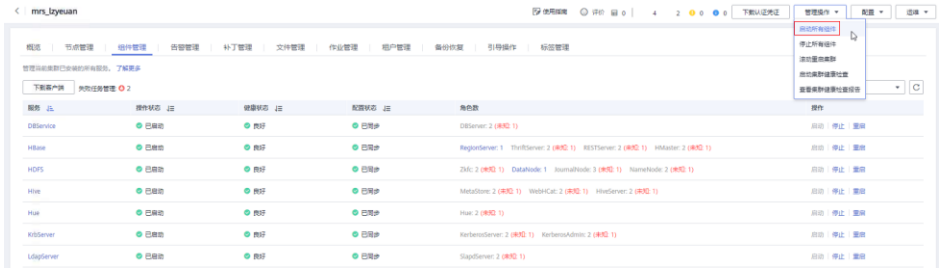
若未同步，可在集群 manager 页面观察到存在异常角色。

原因分析

需解决异常，等集群恢复正常后继续升级 Master 节点。

处理步骤

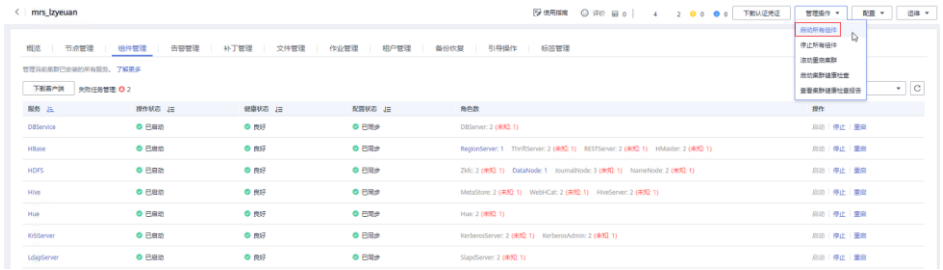
- 方式一：
  a. 进入集群组件管理页，查看服务健康状态与角色状态，若存在“未启动”的角色，进入服务。



  b. 勾选未开启的实例，单击“更多”，选择“启动实例”。

说明

若异常角色较多，可在管理操作处选择启动所有组件。



- 方式二：

  进入集群 manager 页面，查看服务，是否存在“未启动”实例，若存在，启动实例。

- 若存在其他情况导致服务异常无法解决，请联系技术服务协助处理。

2.27 Manager 页面新建的租户删除失败

问题现象

在 FusionInsight Manager 的“租户资源”页面添加租户后，删除租户时，报“删除租户角色失败”。

原因分析

在创建租户时会生成对应的角色，执行删除租户操作时会首先删除对应的角色。此时如果支持权限配置的组件状态异常，则会导致删除这个角色对应的资源权限失败。

处理步骤

步骤 1 登录 FusionInsight Manager，选择“系统 > 权限 > 角色”。

步骤 2 单击“添加角色”，在“配置资源权限”中单击集群名称，确认可配置资源权限的组件。

步骤 3 选择“集群 > 服务”，查看可配置资源权限的组件的运行状态是否都为“良好”。

步骤 4 如果不为“良好”，请启动或者修复组件，直至状态为“良好”。

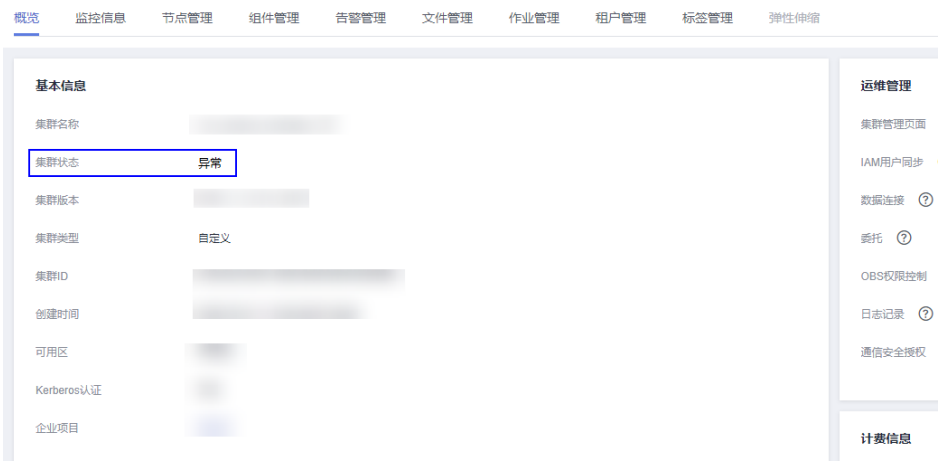
步骤 5 再次执行删除租户操作。

----结束

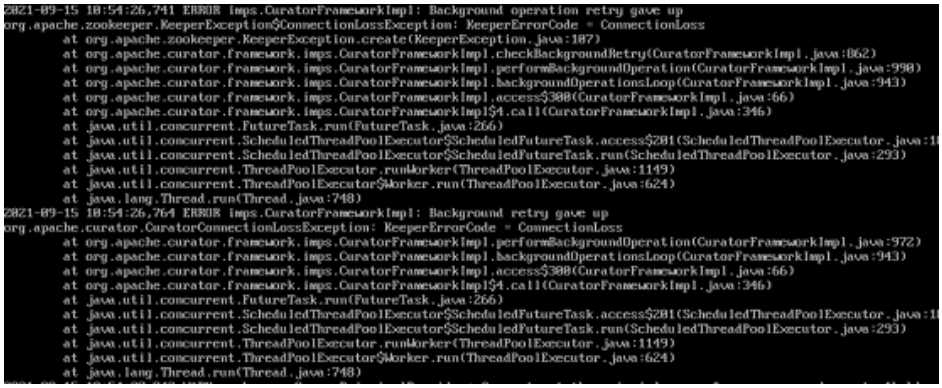
2.28 MRS 集群切换 VPC 后集群状态异常不可用

问题现象

客户 MRS 集群，在 ECS 侧将所有节点的 VPC 切换后集群状态异常。



所有服务不可用，其中 Hive beeline 报错如下：



原因分析

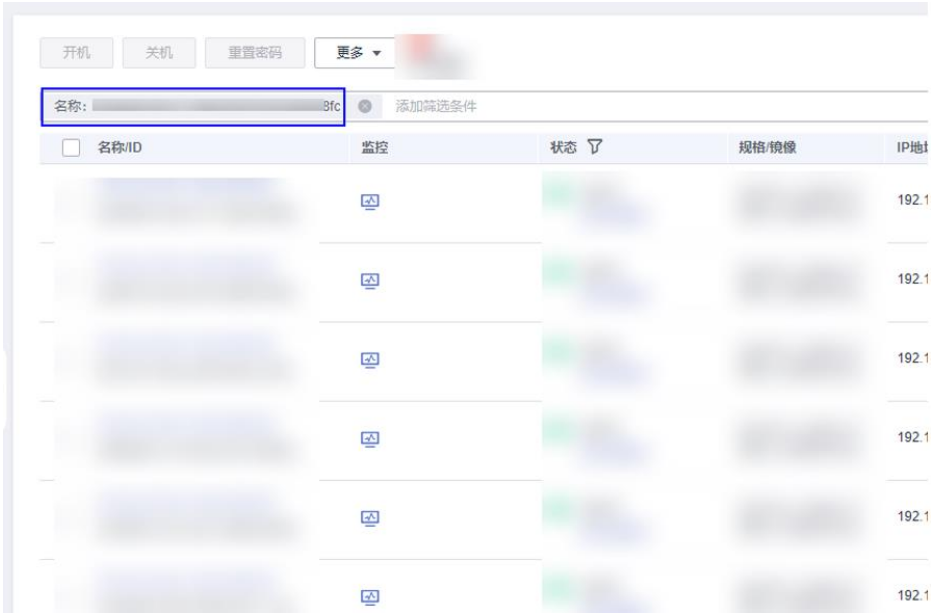
MRS 不支持切换 VPC，切换 VPC 后，节点的内网 IP 变化，但是配置文件和数据库还是原有的 IP，导致集群通信等功能异常，集群状态也会异常。所以要恢复集群，就需要将节点切回到原来的 VPC，并且 IP 和 hosts 文件中的 IP 主机映射一一对应。

处理步骤

步骤 1 登录集群 Master1 节点，执行 **ifconfig** 查看到切换后的 VPC。 执行命令 **cat /etc/hosts**，查看 host 文件中记录切换 VPC 之前的 IP。

步骤 2 登录 MRS 管理控制台，在 MRS 集群概览页面查看“集群 ID”和“虚拟私有云”名称。

步骤 3 登录 ECS 管理控制台，在搜索栏选择“名称”输入 MRS 集群 ID，搜索 MRS 集群的所有节点。



步骤 4 在 MRS 集群节点的操作列选择“更多 > 网络设置 > 切换 VPC”，切换节点 VPC。

说明

- 每个节点都需要切换 VPC。

- 切换 VPC 时“虚拟私有云”、“子网”、“安全组”需要和集群初始的配置一致。
- “私有 IP 地址”选择“现在创建”，填写步骤 1 中查询对应节点的 IP。

步骤 5 切换成功后，单击节点名称，需要在节点的“弹性网卡”中重启“源/目的检查”。

步骤 6 将虚拟 IP 重新绑定到集群的 Master 节点上，操作如下：

1. 登录 MRS 管理控制台，进入 MRS 集群，在“概览”页面单击“前往 Manager”后的 ，“访问方式”选择“专线访问”，记录集群浮动 IP。查看“默认生效子网”后的子网，并记录。
2. 登录 VPC 管理控制台，选择“虚拟私有云 > 子网”，搜索 MRS 集群的子网。
3. 单击子网名称，进入子网详情页面，单击“IP 地址管理”，搜索 MRS 集群浮动 IP。
4. MRS 集群浮动 IP“操作”列的“绑定服务器”，选择 MRS 集群的 Master 节点，绑定成功后如下图：

步骤 7 等待集群恢复。

----结束

2.29 Console 作业管理提交作业异常处理

用户问题

MRS 控制台作业管理提交作业“状态”为“已接受”，“执行结果”为“未定”，作业未成功提交到 Yarn 上，如下图所示：

原因分析

由于 Console 的作业管理功能是由集群管理模块 Executor 来负责调度执行，因此作业未提交到 Yarn 上，根因需要查看 Executor。而 Console 的作业管理功能，正常情况添

加作业后会自动在 Yarn 上启动两个任务，一个提交到 launcher-job 队列，该队列为辅助作业队列。另外一个是作业实际执行的队列，如默认的 default 队列。

查看 Executor 日志（主 Master 节点的“/var/log/executor/exe.log”），发现是由于提交 Flink 作业用户的密码改变或者过期导致下载用户 keytab 认证文件失败，最终导致作业未提交到 launcher-job 队列。

处理步骤

步骤 1 重置提交作业用户的密码。

登录 Manager 页面，选择“系统设置 > 用户管理”。在提交作业的 IAM 用户的操作列，选择“更多 > 初始化密码”，根据界面提示操作。初始化完成后需要使用该用户登录一次 MRS Manager。

步骤 2 登录 MRS 管理控制台，进入 MRS 集群，在“概览”页面，单击“IAM 用户同步”右侧的“单击同步”。

步骤 3 IAM 同步完成，作业管理添加配置作业提交作业即可正常。

----结束

2.30 生成 HA 证书时报错：symbol BIO_new_dgram_sctp version OPENSSL_1_1_0 not defined in file libcrypto.so.1.1 ...

用户问题

更换 HA 证书时，执行 **sh ${OMS_RUN_PATH}/workspace/ha/module/hacom/script/gen-cert.sh --root-ca --country=CN --state=***state* **--city=***city* **--company=***company* **--organize=***organize* **--common-name=***commonname* **--email=***集群用户邮箱*命令在主管理节点“${OMS_RUN_PATH}/workspace0/ha/local/cert”目录生成“root-ca.crt”和“root-ca.pem”时，发生以下报错：

```
openssl: relocation error: openssl: symbol BIO_new_dgram_sctp version OPENSSL_1_1_0 not defined in file libcrypto.so.1.1 with link time referencecreate server private key failed.
```

原因分析

- 用户可能修改了 openssl 不是系统默认的/usr/bin/openssl。
- 动态库依赖 libcrypto.so.1.1 无法找到。
- 如果集群为 3.2.0 及之前版本，可能在执行操作前执行了配置环境变量的命令（例如 **source bigdata_env**），或修改了环境变量。3.2.0 之后版本已修复该问题。

处理步骤

步骤 1 以 **omm** 用户通过主管理节点 IP 登录主管理节点。

步骤 2 执行以下命令查看执行结果是否为“/usr/bin/openssl”。如果不是请修改 openssl 为系统默认的/usr/bin/openssl。

which openssl

步骤 3 执行以下命令查看动态库依赖。

ldd /usr/bin/openssl

例如执行后结果如下：

```
[omm@xxx ~]$ ldd /usr/bin/openssl
        linux-vdso.so.1 (0x0000ffffaf016000)
        libssl.so.1.1 => /usr/lib64/libssl.so.1.1 (0x0000ffffaee7a000)
        libcrypto.so.1.1 => /usr/lib64/libcrypto.so.1.1 (0x0000ffffaebc2000)
        libz.so.1 => /usr/lib64/libz.so.1 (0x0000ffffaeb91000)
        libdl.so.2 => /usr/lib64/libdl.so.2 (0x0000ffffaeb70000)
        libpthread.so.0 => /usr/lib64/libpthread.so.0 (0x0000ffffaeb3b000)
        libc.so.6 => /usr/lib64/libc.so.6 (0x0000ffffae9c5000)
        /lib/ld-linux-aarch64.so.1 (0x0000ffffaefe8000)
```

查看执行结果中 libcrypto.so.1.1 的指向是否有值，如果为 not found 请执行以下命令加载。

echo $LD_LIBRARY_PATH

步骤 4 查看系统库环境变量里是否加载了非系统的 openssl 相关的库。如果是，请修改为系统的 openssl 相关的库。

步骤 5 如果仍旧无法解决，请联系支持人员。

----结束

3 使用 Alluixo

3.1 Alluixo 在 HA 模式下出现 Does not contain a valid host:port authority 报错

用户问题

安全集群 Alluixo 在 HA 模式下出现 Does not contain a valid host:port authority 的报错，如何处理？

问题现象

安全集群中，Alluixo 在 HA 模式下出现 Does not contain a valid host:port authority 的报错。

原因分析

org.apache.hadoop.security.SecurityUtil.buildDTServiceName 不支持在 uri 中填写多个 alluxiomaster 的地址。

处理步骤

使用 alluxio:///或者 alluxio://<主 AlluxioMaster 的 ip 或 hostname>:19998/进行访问。

4 使用 ClickHouse

4.1 ZooKeeper 上数据错乱导致 ClickHouse 启动失败问题

问题现象

ClickHouse 集群中某实例节点启动失败，该实例节点启动日志中有如下类似报错信息：

```
2021.03.15 21:01:19.816593 [ 11111 ] {} <Error> Application: DB::Exception:
The local set of parts of table DEFAULT.lineorder doesn't look like the set
ofdoesn't look like the set of
parts in ZooKeeper: 59.99 million rows of 59.99 million total rows in
filesystem are suspicious. There are 30 unexpected parts with 59986052 rows
(14 of them is not just-written with 59986052 rows), 0 missing parts (with 0
blocks).: Cannot attach table `DEFAULT`.`lineorder` from metadata file
...
: while loading database
```

原因分析

使用 ClickHouse 过程中，ClickHouse 实例异常场景下，重复创建集群 ReplicatedMergeTree 引擎表，后续又进行删除表等操作导致 ZooKeeper 上的数据异常，致使 ClickHouse 启动失败。

解决办法

步骤 1 备份问题节点数据库下所有表数据到其他目录。

- 备份表数据：

  MRS 3.0.5 及之前版本

  cd /srv/BigData/data1/clickhouse/data/*数据库名*

  mkdir -p *备份目录***/data1**

  mv *{表名} 备份目录***/data1/**

  MRS 3.1.0 及之后版本

  head -1 /srv/BigData/data1/clickhouse_path/metadata/*库名|表名***.sql | awk -F ' ' '{print $5}' | sed "s/'//g"** *获取到目标的 UUID*

cd /srv/BigData/data1/clickhouse/store/*{UUID 前 3 个字符}*

mkdir -p *备份目录***/data1**

mv *{UUID}* *备份目录***/data1/**

说明

如果存在多磁盘的场景，需要对 **data1** 到 **dataN** 的磁盘数据都执行相同的备份操作。

- 备份元数据信息：

  cd /srv/BigData/data1/clickhouse_path/metadata/库名

  mv *表名***.sql** *备份目录*

例如，下面是备份 default 数据库下的表 lineorder 数据到**/home/backup** 目录下。

cd /srv/BigData/data1/clickhouse/data/default

mkdir -p /home/backup/data1

mv lineorder /home/backup/data1

cd /srv/BigData/data1/clickhouse_path/metadata

mv lineorder.sql /home/backup

步骤 2 登录 MRS Manager 页面，选择“集群 > 服务 > ClickHouse > 实例”，选择对应的实例节点，单击“启动实例”，完成实例启动。

步骤 3 实例启动成功后，使用 ClickHouse 客户端登录问题节点。

clickhouse client --host *clickhouse 实例 IP* **--user** *用户名* **--password** *密码*

步骤 4 执行以下命令获取当前表所在的 ZooKeeper 路径 **zookeeper_path** 和对应节点所在的副本编号 **replica_num**。

SELECT zookeeper_path FROM system.replicas WHERE database = '*数据库名***' AND table = '***表名***';**

SELECT replica_num,host_name FROM system.clusters;

步骤 5 执行以下命令连接 ZooKeeper 命令行界面。

zkCli.sh -server *ZooKeeper 所在节点的 IP*:**2181**

步骤 6 找到对应故障节点表数据对应的 ZooKeeper 路径。

ls *zookeeper_path*/**replicas/***replica_num*

说明

zookeeper_path 为步骤 4 中查询到的 **zookeeper_path** 值。

replica_num 为步骤 4 中节点主机对应的副本编号 **replica_num** 的值。

步骤 7 执行以下命令，删除 ZooKeeper 上的副本数据。

deleteall *zookeeper_path*/**replicas/***replica_num*

步骤 8 使用 ClickHouse 客户端登录问题节点，重新执行 create 创建集群 ReplicatedMergeTree 引擎表。

clickhouse client --host *clickhouse 实例IP* **--multiline --user** *用户名* **--password** *密码*

CREATE TABLE *数据库名.表名* **ON CLUSTER** *集群名*

...

ENGINE = ReplicatedMergeTree ...

其他副本节点有如下提示表已经存在的报错信息，属于正常现象，可以忽略。

```
Received exception from server (version 20.8.7):
Code: 57. DB::Exception: Received from x.x.x.x:9000. DB::Exception:
There was an error on [x.x.x.x:9000]: Code: 57, e.displayText() =
DB::Exception: Table DEFAULT.lineorder already exists. (version 20.8.11.17
(official build)).
```

建表成功后问题节点上表数据会自动进行同步，数据恢复完成。

----结束

4.2 ClickHouse 消费 Kafka 数据异常

问题现象

用户在 ClickHouse 集群创建 Kafka 引擎表 test.user_log_kafka 消费 Kafka 数据，查看 Kafka 监控发现凌晨开始出现消息堆积，数据一直没有被消费。

原因分析

Kafka 出现消息堆积，说明 ClickHouse 消费数据时出现异常，需要看一下 ClickHouse 的日志。

1. 登录 MRS 集群，进入 ClickHouse 实例所在的节点，查看“/var/log/Bigdata/clickhouse”目录下的“clickhouse-server.log”日志文件，发现以下报错：

2. 进入到其他 ClickHouse 节点也发现了同样的报错日志，由此可知 Kafka 消息堆积是因为 ClickHouse 解析 Kafka 数据时出现异常。

解决办法

使用以下命令修改表的“kafka_skip_broken_messages”属性：

```
# ALTER test.user_log MODIFY SETTINGS kafka_skip_broken_messages=10000
```

说明

- 10000 可以根据数据中脏数据的比例进行调整。
- kafka_skip_broken_messages：Kafka 消息解析器对每个块的架构不兼容消息的容忍度，默认值：0。

  例如：kafka_skip_broken_messages = N，则引擎会跳过 N 条无法解析的 Kafka 消息。

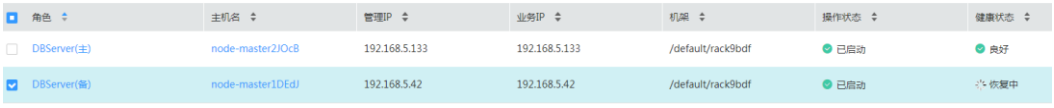
5 使用 DBservice

5.1 DBServer 实例状态异常

问题背景与现象

DBServer 实例状态一直是 concerning。

图5-1 DBServer 实例状态



角色	主机名	管理IP	业务IP	机架	操作状态	健康状态
DBServer(主)	node-master2JOcB	192.168.5.133	192.168.5.133	/default/rack9bdf	已启动	良好
DBServer(备)	node-master1DEdJ	192.168.5.42	192.168.5.42	/default/rack9bdf	已启动	恢复中

原因分析

数据目录下文件或目录的权限不对，GaussDB 要求文件权限至少是 600，目录权限至少为 700。

图5-2 目录权限列表

```
omm@ 192-168-234-176:/srv/BigData/dbdata_service> ll
total 4
drwx------ 19 omm wheel 4096 Dec 14 10:15 data
```

图5-3 文件权限列表

```
omm@192-168-234-176:/srv/BigData/dbdata_service/data> ll
total 128
drwx------ 6 omm wheel  4096 Dec  9 15:47 base
-rw------- 1 omm wheel   922 Dec  9 15:34 dblink.conf
-rw------- 1 omm wheel    16 Dec 14 10:15 gaussdb.state
drwx------ 2 omm wheel  4096 Dec 14 10:17 global
drwx------ 2 omm wheel  4096 Dec 11 00:00 pg_audit
drwx------ 2 omm wheel  4096 Dec 14 10:15 pg_blackbox
drwx------ 2 omm wheel  4096 Dec  9 15:34 pg_clog
drwx------ 2 omm wheel  4096 Dec 14 10:15 pg_confile_backup
-rw------- 1 omm wheel  1024 Dec  9 15:34 pg_ctl.lock
-rw------- 1 omm wheel  4245 Dec  9 15:47 pg_hba.conf
-rw------- 1 omm wheel  1024 Dec  9 15:47 pg_hba.conf.lock
-rw------- 1 omm wheel  1636 Dec  9 15:34 pg_ident.conf
drwx------ 2 omm wheel  4096 Dec  9 15:38 pg_log
drwx------ 4 omm wheel  4096 Dec  9 15:34 pg_multixact
drwx------ 2 omm wheel  4096 Dec 14 10:15 pg_notify
drwx------ 2 omm wheel  4096 Dec  9 15:34 pg_serial
drwx------ 2 omm wheel  4096 Dec  9 15:34 pg_snapshots
drwx------ 2 omm wheel  4096 Dec 14 11:56 pg_stat_tmp
drwx------ 2 omm wheel  4096 Dec  9 15:34 pg_subtrans
drwx------ 2 omm wheel  4096 Dec  9 15:34 pg_tblspc
drwx------ 2 omm wheel  4096 Dec  9 15:34 pg_twophase
-rw------- 1 omm wheel     4 Dec  9 15:34 PG_VERSION
drwx------ 2 omm wheel  4096 Dec  9 15:34 pg_wallet
drwx------ 3 omm wheel  4096 Dec  9 15:39 pg_xlog
-rw------- 1 omm wheel 13309 Dec 14 10:15 postgresql.conf
-rw------- 1 omm wheel  1024 Dec  9 15:34 postgresql.conf.lock
-rw------- 1 omm wheel   105 Dec 14 10:15 postmaster.opts
-rw------- 1 omm wheel    96 Dec 14 10:15 postmaster.pid
```

解决办法

步骤 1　按照图 5-2 和图 5-3 的权限列表，修改相应文件和目录的权限。

步骤 2　重启相应的 DBServer 实例。

----结束

5.2 DBServer 实例一直处于 Restoring 状态

问题背景与现象

DBServer 实例状态一直是 Restoring 状态，重启之后仍然不恢复。

原因分析

1. DBService 组件会对“${BIGDATA_HOME}/MRS_*XXX*/install/dbservice/ha/module/harm/plugin/script/gsDB/.startGS.fail”这个文件监控。其中 *XXX* 是产品版本号。

2. 如果这个文件中的值大于 3 就会启动失败，NodeAgent 会一直尝试重启该实例，此时仍会失败而且这个值每启动失败一次就会加 1。

解决办法

步骤 1 登录 Manager 管理界面。

步骤 2 停止该 DBServer 实例。

步骤 3 使用 **omm** 用户登录到 DBServer 实例异常的节点。

步骤 4 修改“${BIGDATA_HOME}/MRS_*XXX*/install/dbservice/ha/module/harm/plugin/script/gsDB/.startGS.fail”配置文件中的值为 0。其中 *XXX* 是产品版本号。

步骤 5 启动该 DBServer 实例。

----结束

5.3 默认端口 20050 或 20051 被占用

问题背景与现象

执行 DBService 服务重启操作时，DBService 服务启动失败，打印的错误日志中出现 20050 或 20051 端口被占用等信息。

原因分析

1. 由于 DBService 使用的默认端口 20050 或 20051 被其他进程占用。
2. DBService 进程没有停止成功，使用的端口未释放。

解决办法

该解决办法以 20051 端口被占用为例，20050 端口被占用的解决办法与该办法类似。

步骤 1 以 **root** 用户登录 DBService 安装报错的节点主机，执行命令：**netstat -nap | grep 20051** 查看占用 20051 端口的进程。

步骤 2 使用 **kill** 命令强制终止使用 20051 端口的进程。

步骤 3 约 2 分钟后，再次执行命令：**netstat -nap | grep 20051**，查看是否还有进程占用该端口。

步骤 4 确认占用该端口进程所属的服务，并修改为其他端口。

步骤 5 分别在“/tmp”和“/var/run/MRS-DBService/”目录下执行 **find . -name "*20051*"**命令，将搜索到的文件全部删除。

步骤 6 登录 Manager，重启 DBService 服务。

----结束

5.4 /tmp 目录权限不对导致 DBserver 实例状态一直处于 Restoring

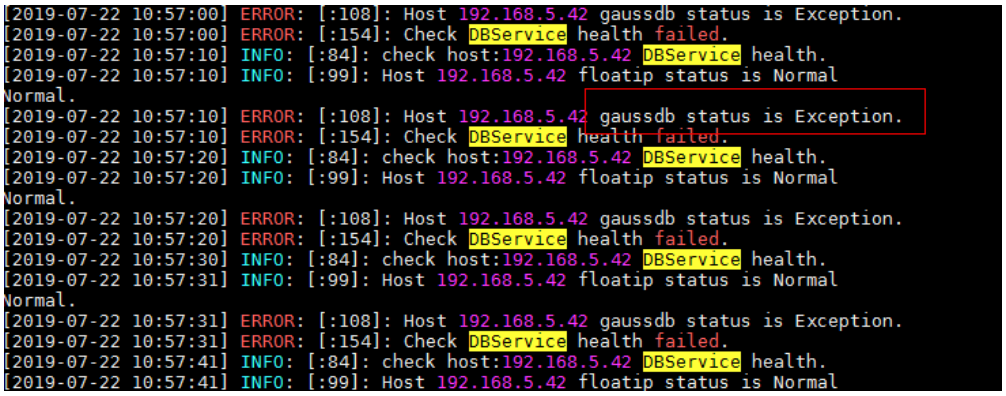
问题背景与现象

DBServer 实例状态一直是 Restoring 状态，重启之后仍然不恢复。

原因分析

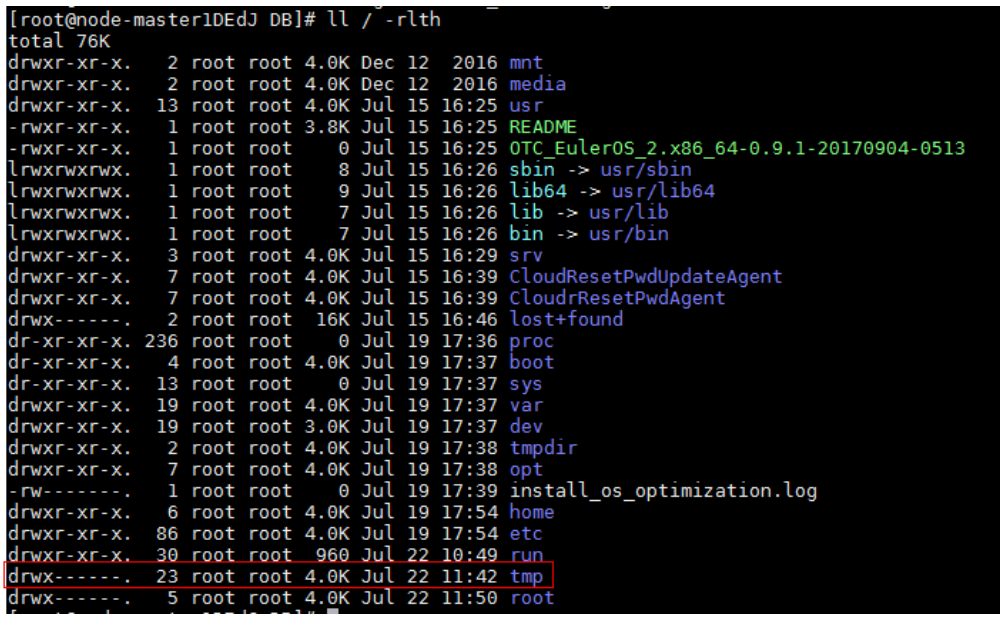
1. 查看“/var/log/Bigdata/dbservice/healthCheck/dbservice_processCheck.log”，可以看到 gaussdb 异常。

   图5-4 gaussdb 异常



2. 检查发现“/tmp”权限不对。

图5-5 /tmp 权限



解决办法

步骤 1　修改/tmp 的权限。

chmod 1777 /tmp

步骤 2　等待实例状态恢复。

----结束

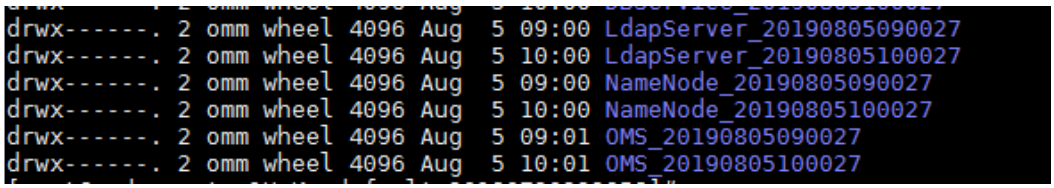
5.5 DBService 备份失败

问题背景与现象

ls /srv/BigData/LocalBackup/default_20190720222358/ -rlth

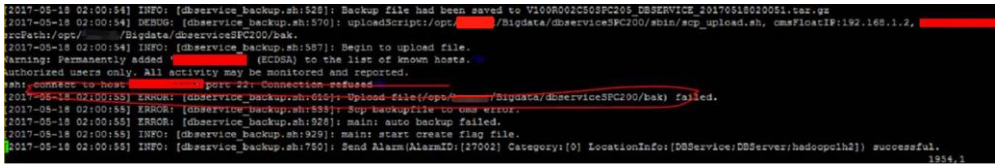
查看备份文件路径中没有 DBService 的备份文件。

图5-6 查看备份文件



原因分析

- 查看 DBService 的备份日志/var/log/Bigdata/dbservice/scriptlog/backup.log，其实备份已经成功，只是上传至 OMS 节点时失败。



- 失败原因是由于 ssh 不通。

解决办法

步骤 1 网络问题，联系网络工程师处理。

步骤 2 网络问题解决之后重新备份即可。

----结束

5.6 DBService 状态正常，组件无法连接 DBService

问题背景与现象

上层组件连接 DBService 失败，检查 DBService 组件状态正常，两个实例状态也正常。

图5-7 DBService 状态

原因分析

1. 上层组件是通过 dbservice.floatip 连接的 DBService。
2. 在 DBServer 所在节点执行命令 netstat -anp | grep 20051 发现，DBService 的 Gauss 进程在启动时并未绑定 floatip，只监听了 127.0.0.1 的本地 ip。

解决办法

步骤 1 重新启动 DBService 服务。

步骤 2　启动完成之后在主 DBServer 节点执行 **netstat -anp | grep 20051** 命令检查是否绑定了 dbservice.floatip。

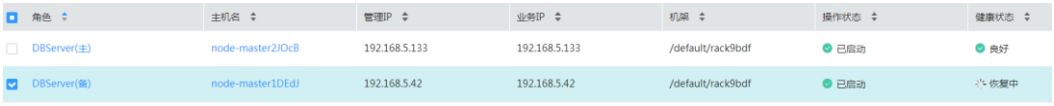
----结束

5.7 DBServer 启动失败

问题背景与现象

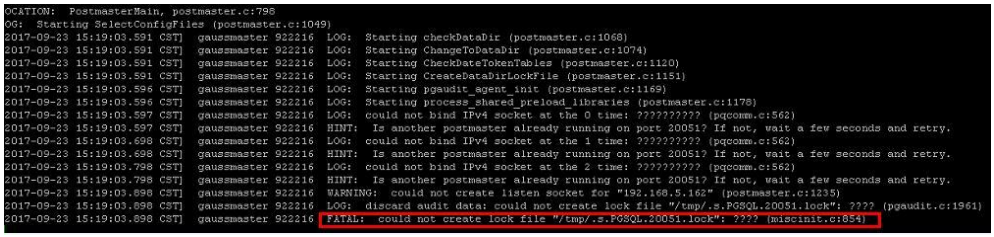
DBService 组件启动失败，重启还是失败，实例状态一直为正在恢复状态。

图5-8 DBService 的状态

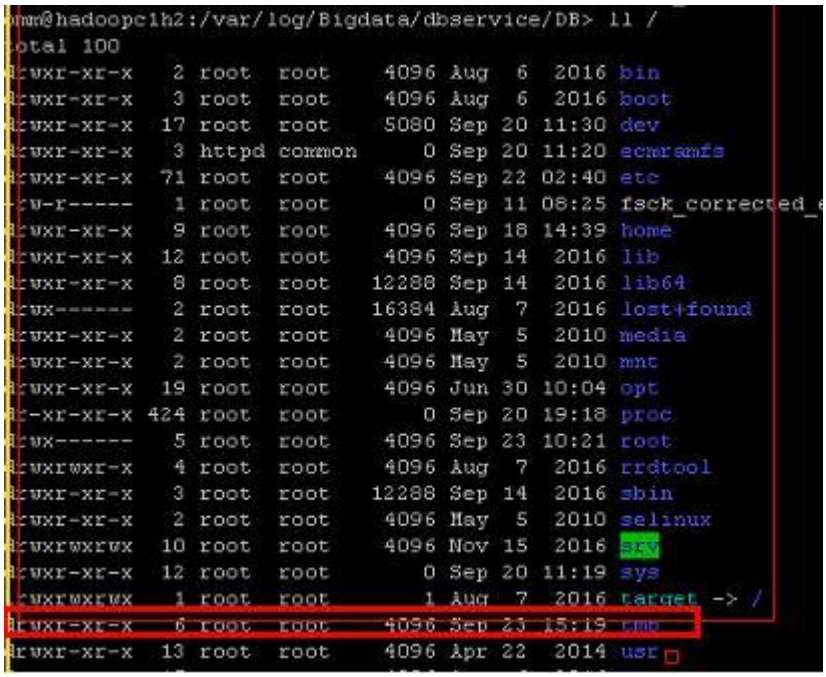


原因分析

1. 查看 DBService 的日志/var/log/Bigdata/dbservice/DB/gs_ctl-current.log，报如下错误。



2. 检查发现/tmp 权限不正确，正确的权限应该为 777。



解决办法

步骤 1　修改/tmp 权限为 777。

步骤 2　重新启动 DBService 组件。

----结束

5.8 浮动 IP 不通导致 DBService 备份失败

问题背景与现象

在默认备份 default 中 DBService 备份失败，其他备份（NameNode、LdapServer、OMS 备份）成功。

原因分析

1. 查看 DBService 的备份页面错误信息，有如下错误信息提示：

```
Clear temporary files at backup checkpoint
DBService_test_DBService_DBService_20180326155921 that fialed last time.
Temporyary files at backup checkpoint
DBService_test_DBService_DBService20180326155921 that failed last time are
cleared successfully.
```

```
Start executing the backup task.
The backup of configuration DBService is started.
Check the backup available disk space.
Backup initialization succeeded for configuration DBService.
Clear temporary files at backup checkpoint DBService_test_DBService_DBService_20180326155921 that failed last time.
Temporary files at backup checkpoint DBService_test_DBService_DBService_20180326155921 that failed last time are cleared successfully.
Checkpoint DBService_test_DBService_DBService_20180326162235 is verified successfully before backup.
Temporary files are cleared successfully before backup checkpoint DBService_test_DBService_DBService_20180326162235.
Prestart backup succeeded for checkpoint DBService_test_DBService_DBService_20180326162235.
The snapshot is created successfully for checkpoint DBService_test_DBService_DBService_20180326162235 before backup.
Backup is being performed for checkpoint DBService_test_DBService_DBService_20180326162235.
Backup execution failed. Task ID: 2
Detail: DBService backup task failed, please view details in logs.
Temporary files are cleared successfully after backup checkpoint DBService_test_DBService_DBService_20180326162235.
checkpoint DBService_test_DBService_DBService_20180326162235 is deleted successfully after backup failure.
Failed to backup configuration DBService.
```

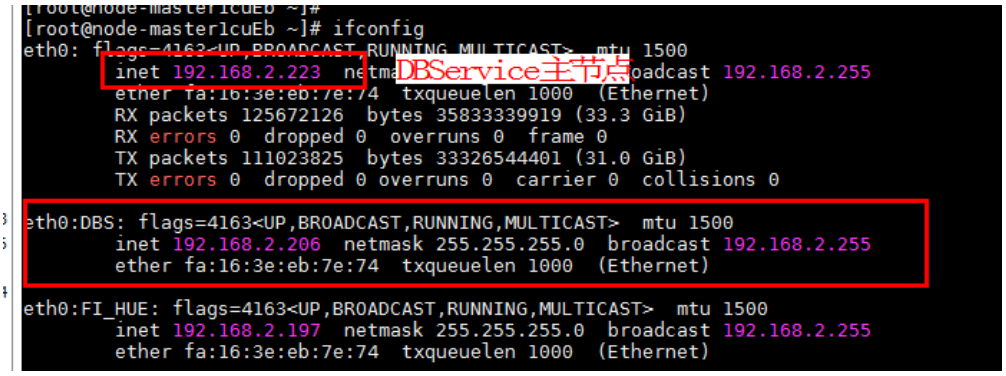
2. 查看/var/log/Bigdata/dbservice/scriptlog/backup.log 文件，发现日志停止打印，并没有备份相关信息。
3. 查看主 OMS 节点 /var/log/Bigdata/controller/backupplugin.log 日志发现如下错误信息：

```
result error is ssh:connect to host 172.16.4.200 port 22 : Connection refused
(172.16.4.200 是 DBService 的浮动 IP)
DBService backup failed.
```

```
2018-03-27 07:00:35,758 INFO  [pool-1-thread-5] Create adapter from com.[redacted].bigdata.om.backup.MetadataPluginAdapter success.
com.[redacted].bigdata.om.backup.plugin.AbstractBackupRecoveryPlugin.initializePluginAdapter(AbstractBackupRecoveryPlugin.java:92)
2018-03-27 07:00:35,759 INFO  [pool-1-thread-5] floatIp is 172.16.4.200. com.[redacted].bigdata.om.dbservice.backup.BackupRecoveryPlugin.getFloatIp(BackupRecoveryPlugin.java:233)
2018-03-27 07:00:35,759 INFO  [pool-1-thread-5] cmd is ssh 172.16.4.200 /opt/[redacted]/Bigdata/FusionInsight_V100R002C60U20/dbservice/sbin/dbservice_backup.sh -b -d
/srv/BigData/LocalBackup/default_20180326213206/DBService_20180327070010. com.[redacted].bigdata.om.dbservice.backup.BackupRecoveryPlugin.startBackup(BackupRecoveryPlugin.java:166)
2018-03-27 07:00:35,759 INFO  [pool-1-thread-5] create task taskId is 6. com.[redacted].bigdata.om.dbservice.backup.BackupRecoveryPlugin.startBackup(BackupRecoveryPlugin.java:169)
2018-03-27 07:00:35,760 INFO  [pool-1-thread-5] startBackup result OperateResult(errorCode:RUNNING, result:6, detailInfo:, packageName:null).
com.[redacted].bigdata.om.backup.BackupPluginContainerHandler.startBackup(BackupPluginContainerHandler.java:246)
2018-03-27 07:00:35,760 INFO  [Thread-132] Executing the command with arguments and env, timeout: 900000
com.[redacted].bigdata.om.controller.api.extern.monitor.script.LinuxScriptExecutionHandler.logMessage(LinuxScriptExecutionHandler.java:64)
2018-03-27 07:00:35,863 INFO  [Thread-132] Execute command : /opt/[redacted]Bigdata/om-0.0.1/sbin/scriptLauncher.sh ssh 172.16.4.200
/opt/[redacted]/Bigdata/FusionInsight_V100R002C60U20/dbservice/sbin/dbservice_backup.sh -b -d /srv/BigData/LocalBackup/default_20180326213206/DBService_20180327070010.
com.[redacted].bigdata.om.dbservice.backup.BackupTask.run(BackupTask.java:48)
2018-03-27 07:00:35,863 INFO  [Thread-132] result status is 255. com.[redacted].bigdata.om.dbservice.backup.BackupTask.run(BackupTask.java:49)
2018-03-27 07:00:35,863 INFO  [Thread-132] result output is . com.[redacted].bigdata.om.dbservice.backup.BackupTask.run(BackupTask.java:50)
2018-03-27 07:00:35,863 INFO  [Thread-132] result erro is ssh: connect to host 172.16.4.200 port 22: Connection refused
. com.[redacted].bigdata.om.dbservice.backup.BackupTask.run(BackupTask.java:51)
2018-03-27 07:00:35,863 ERROR [Thread-132] DBService backup failed. com.[redacted].bigdata.om.dbservice.backup.BackupTask.run(BackupTask.java:64)
2018-03-27 07:00:40,868 INFO  [pool-1-thread-5] query backup taskId is 6. com.[redacted].bigdata.om.dbservice.backup.BackupRecoveryPlugin.getBackupProgress(BackupRecoveryPlugin.java:247)
```

解决办法

步骤 1　登录 DBService 主节点（绑定有 DBService 浮动 IP 的 master 节点）。



步骤 2　检查 /etc/ssh/sshd_config 文件中 ListenAddress 配置项，添加 DBService 浮动 IP 到 ListenAdress 或者注释掉 ListenAdress 配置项。

步骤 3　执行如下命令重启 sshd 服务。

service sshd restart

步骤 4　观察下次备份 DBServie 是否备份成功。

----结束

5.9 DBService 配置文件丢失导致启动失败

问题背景与现象

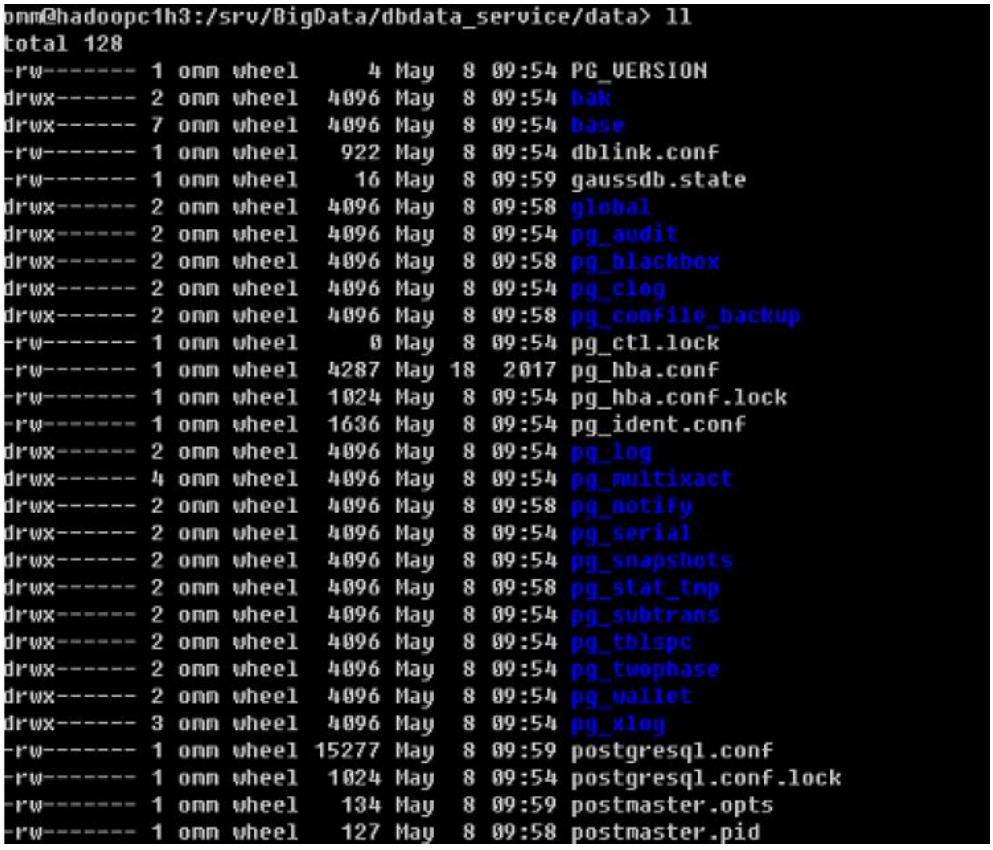
节点异常下电，重启备 DBService 失败。

原因分析

1. 查看/var/log/Bigdata/dbservice/DB/gaussdb.log 日志没有内容。
2. 查看/var/log/Bigdata/dbservice/scriptlog/preStartDBService.log 日志，发现如下信息，判断为配置信息丢失。

```
The program "gaussdb" was found by "
/opt/Bigdata/MRS_xxx/install/dbservice/gaussdb/bin/gs_guc)
But not was not the same version as gs_guc.
Check your installation.
```

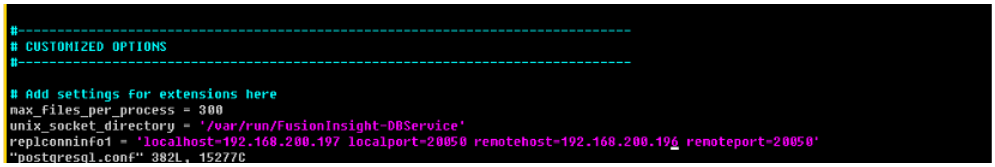
3. 比对主备 DBServer 节点/srv/BigData/dbdata_service/data 目录下的配置文件发现差距比较大。



解决办法

步骤 1 把主节点**/srv/BigData/dbdata_service/data** 的内容拷贝到备节点，保持文件权限和属组与主节点一样。

步骤 2 修改 postgresql.conf 配置信息，localhost 修改成本节点 IP，remotehost 修改成对端节点 IP。



步骤 3　登录 Manager 页面重启备 DBServer 节点。

----结束

6 使用 Flink

6.1 安装客户端执行命令错误，提示 IllegalConfigurationException: Error while parsing YAML configuration file :"security.kerberos.login.keytab"

问题背景与现象

客户端安装成功，执行客户端命令例如 yarn-session.sh 命令报错，提示 IllegalConfigurationException: Error while parsing YAML configuration file : "security.kerberos.login.keytab: "

```
[root@8-5-131-10 bin]# yarn-session.sh
2018-10-25 01:22:06,454 | ERROR | [main] | Error while trying to split key and
value in configuration file /opt/flinkclient/Flink/flink/conf/flink-conf.yaml:80:
"security.kerberos.login.keytab: " |
org.apache.flink.configuration.GlobalConfiguration (GlobalConfiguration.java:160)
Exception in thread "main"
org.apache.flink.configuration.IllegalConfigurationException: Error while parsing
YAML configuration file :80: "security.kerberos.login.keytab: "
       at
org.apache.flink.configuration.GlobalConfiguration.loadYAMLResource(GlobalConfigura
tion.java:161)
       at
org.apache.flink.configuration.GlobalConfiguration.loadConfiguration(GlobalConfigur
ation.java:112)
       at
org.apache.flink.configuration.GlobalConfiguration.loadConfiguration(GlobalConfigur
ation.java:79)
       at
org.apache.flink.yarn.cli.FlinkYarnSessionCli.main(FlinkYarnSessionCli.java:482)
[root@8-5-131-10 bin]#
```

原因分析

在安全集群环境下，Flink 需要进行安全认证。当前客户端未进行相关安全认证设置。

1. Flink 整个系统有两种认证方式：

- 使用 kerberos 认证：Flink yarn client、Yarn Resource Manager、JobManager、HDFS、TaskManager、Kafka 和 Zookeeper。
- 使用 YARN 内部的认证机制：Yarn Resource Manager 与 Application Master（简称 AM）。

2. 如果用户安装安全集群需要使用 kerberos 认证和 security cookie 认证。根据日志提示，发现配置文件中“security.kerberos.login.keytab :”配置项错误，未进行安全配置。

解决办法

步骤 1 从 MRS 上下载用户 keytab，并将 keytab 放到 Flink 客户端所在主机的某个文件夹下。

步骤 2 在“flink-conf.yaml”上配置：

1. keytab 路径。

```
security.kerberos.login.keytab: /home/flinkuser/keytab/abc222.keytab
```

说明

- “/home/flinkuser/keytab/abc222.keytab”表示的是用户目录，为步骤 1 中放置目录。
- 请确保客户端用户具备对应目录权限。

2. principal 名。

```
security.kerberos.login.principal: abc222
```

3. 对于 HA 模式，如果配置了 ZooKeeper，还需要设置 ZK kerberos 认证相关的配置。配置如下：

```
zookeeper.sasl.disable: false
security.kerberos.login.contexts: Client
```

4. 如果用户对于 Kafka client 和 Kafka broker 之间也需要做 kerberos 认证，配置如下：

```
security.kerberos.login.contexts: Client,KafkaClient
```

----结束

6.2 安装客户端修改配置后执行命令错误，提示 IllegalConfigurationException: Error while parsing YAML configuration file

问题背景与现象

客户端安装成功，执行客户端命令例如 yarn-session.sh 命令报错，提示 IllegalConfigurationException: Error while parsing YAML configuration file :81: "security.kerberos.login.principal:pippo "

```
[root@8-5-131-10 bin]# yarn-session.sh
2018-10-25 19:27:01,397 | ERROR | [main] | Error while trying to split key and
value in configuration file /opt/flinkclient/Flink/flink/conf/flink-conf.yaml:81:
"security.kerberos.login.principal:pippo " |
org.apache.flink.configuration.GlobalConfiguration (GlobalConfiguration.java:160)
```

```
Exception in thread "main"
org.apache.flink.configuration.IllegalConfigurationException: Error while parsing
YAML configuration file :81: "security.kerberos.login.principal:pippo "
       at
org.apache.flink.configuration.GlobalConfiguration.loadYAMLResource(GlobalConfigura
tion.java:161)
       at
org.apache.flink.configuration.GlobalConfiguration.loadConfiguration(GlobalConfigur
ation.java:112)
       at
org.apache.flink.configuration.GlobalConfiguration.loadConfiguration(GlobalConfigur
ation.java:79)
       at
org.apache.flink.yarn.cli.FlinkYarnSessionCli.main(FlinkYarnSessionCli.java:482)
```

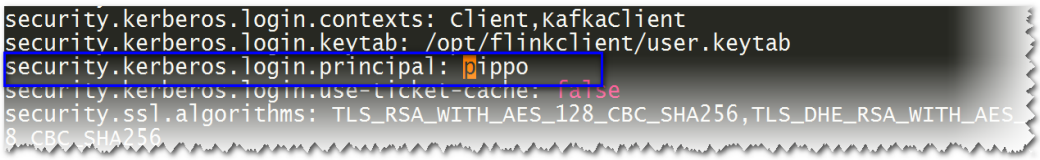
原因分析

配置文件 flink-conf.yaml 中配置项"security.kerberos.login.principal:pippo”错误。

解决办法

修改 flink-conf.yaml 中配置。

注意：配置项名称和值之间存在空格。



6.3 创建 Flink 集群时执行 yarn-session.sh 命令失败

问题背景与现象

创建 Fllink 集群时，执行 yarn-session.sh 命令卡住一段时间后报错：

```
2018-09-20 22:51:16,842 | WARN  | [main] | Unable to get ClusterClient status from
Application Client | org.apache.flink.yarn.YarnClusterClient
(YarnClusterClient.java:253)
org.apache.flink.util.FlinkException: Could not connect to the leading JobManager.
Please check that the JobManager is running.
    at
org.apache.flink.client.program.ClusterClient.getJobManagerGateway(ClusterClient.ja
va:861)
    at
org.apache.flink.yarn.YarnClusterClient.getClusterStatus(YarnClusterClient.java:248)
```

```
    at
org.apache.flink.yarn.YarnClusterClient.waitForClusterToBeReady(YarnClusterClient.j
ava:516)
    at
org.apache.flink.yarn.cli.FlinkYarnSessionCli.run(FlinkYarnSessionCli.java:717)
    at
org.apache.flink.yarn.cli.FlinkYarnSessionCli$1.call(FlinkYarnSessionCli.java:514)
    at
org.apache.flink.yarn.cli.FlinkYarnSessionCli$1.call(FlinkYarnSessionCli.java:511)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:422)
    at
org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1729)
    at
org.apache.flink.runtime.security.HadoopSecurityContext.runSecured(HadoopSecurityCo
ntext.java:41)
    at
org.apache.flink.yarn.cli.FlinkYarnSessionCli.main(FlinkYarnSessionCli.java:511)
Caused by: org.apache.flink.runtime.leaderretrieval.LeaderRetrievalException: Could
not retrieve the leader gateway.
    at
org.apache.flink.runtime.util.LeaderRetrievalUtils.retrieveLeaderGateway(LeaderRetr
ievalUtils.java:79)
    at
org.apache.flink.client.program.ClusterClient.getJobManagerGateway(ClusterClient.ja
va:856)
    ... 10 common frames omitted
Caused by: java.util.concurrent.TimeoutException: Futures timed out after [10000
milliseconds]
```

可能原因

Flink 开启了 SSL 通信加密，却没有正确的配置 SSL 证书。

解决办法

针对 MRS 2.x 及之前版本，操作如下：

方法 1：

关闭 Flink SSL 通信加密，修改客户端配置文件 **conf/flink-conf.yaml**。

```
security.ssl.internal.enabled: false
```

方法 2：

开启 Flink SSL 通信加密，security.ssl.internal.enabled 保持默认。正确配置 SSL：

- 配置 keystore 或 truststore 文件路径为**相对路径**时，Flink Client 执行命令的目录需要可以直接访问该相对路径

  ```
  security.ssl.internal.keystore: ssl/flink.keystore
  security.ssl.internal.truststore: ssl/flink.truststore
  ```

  在 Flink 的 CLI yarn-session.sh 命令中增加“-t”选项来传输 keystore 和 truststore 文件到各个执行节点。如：

```
yarn-session.sh -t ssl/ 2
```

- 配置 keystore 或 truststore 文件路径为**绝对路径**时，需要在 Flink Client 以及各个节点的该绝对路径上放置 keystore 或 truststore 文件。

```
security.ssl.internal.keystore: /opt/client/Flink/flink/conf/flink.keystore
security.ssl.internal.truststore: /opt/client/Flink/flink/conf/flink.truststore
```

针对 MRS3.x 及之后版本，操作如下：

方法 1：

关闭 Flink SSL 通信加密，修改客户端配置文件 **conf/flink-conf.yaml**。

```
security.ssl.enabled: false
```

方法 2：

开启 Flink SSL 通信加密，security.ssl.enabled 保持默认。正确配置 SSL：

- 配置 keystore 或 truststore 文件路径为**相对路径**时，Flink Client 执行命令的目录需要可以直接访问该相对路径

```
security.ssl.keystore: ssl/flink.keystore
security.ssl.truststore: ssl/flink.truststore
```

在 Flink 的 CLI yarn-session.sh 命令中增加“-t”选项来传输 keystore 和 truststore 文件到各个执行节点。如：

```
yarn-session.sh -t ssl/ 2
```

- 配置 keystore 或 truststore 文件路径为**绝对路径**时，需要在 Flink Client 以及各个节点的该绝对路径上放置 keystore 或 truststore 文件。

```
security.ssl.keystore: /opt/client/Flink/flink/conf/flink.keystore
security.ssl.truststore: /opt/client/Flink/flink/conf/flink.truststore
```

6.4 使用不同用户，执行 yarn-session 创建集群失败

问题背景与现象

使用 Flink 过程中，具有两个相同权限用户 testuser 和 bdpuser。

使用用户 testuser 创建 Flink 集群正常，但是切换至 bdpuser 用户创建 Fllink 集群时，执行 yarn-session.sh 命令报错：

```
2019-01-02 14:28:09,098 | ERROR | [main] | Ensure path threw exception |
org.apache.flink.shaded.curator.org.apache.curator.framework.imps.CuratorFrameworkI
mpl (CuratorFrameworkImpl.java:566)
org.apache.flink.shaded.zookeeper.org.apache.zookeeper.KeeperException$NoAuthExcept
ion: KeeperErrorCode = NoAuth for /flink/application_1545397824912_0022
```

可能原因

高可用配置项未修改。由于在 Flink 的配置文件中，**high-availability.zookeeper.client.acl** 默认为 **creator**，仅创建者有权限访问，新用户无法访问 ZooKeeper 上的目录导致 yarn-session.sh 执行失败。

解决办法

步骤 1 修改客户端配置文件 **conf/flink-conf.yaml** 中配置项 **high-availability.zookeeper.path.root**，例如：

```
high-availability.zookeeper.path.root：  flink2
```

步骤 2 重新提交任务。

----结束

6.5 Flink 业务程序无法读取 NFS 盘上的文件

用户问题

Flink 业务程序无法读取集群节点挂载的 NFS 盘上的文件。

问题现象

用户开发的 Flink 业务程序中需要读取用户定义的配置文件，该配置文件放在 NFS 盘上，NFS 盘是挂载在集群节点上的，集群的所有节点均可以访问该盘。用户提交 flink 程序后，业务代码访问不到客户自定义的配置文件，导致业务程序启动失败。

原因分析

该问题的根因是 NFS 盘上的根目录权限不足，导致 Flink 程序启动后无法访问该目录。

MRS 的 Flink 任务是在 YARN 运行，当集群为普通集群时，在 YARN 上运行任务的用户为 yarn_user。用户自定义的配置文件如果在任务启动之后使用，则文件以及文件的父目录（NFS 上的文件所在的父目录，非集群节点上的软连接），必须允许 yarn_user 可以访问，否则程序中无法获取文件内容。当集群为 kerberos 集群时，则文件的权限必须允许提交程序的用户访问。

处理步骤

步骤 1 以 root 用户登录集群的 Master 节点。

步骤 2 执行如下命令查看用户自定义配置文件所在父目录的权限。

ll <文件所在路径的父目录路径>

步骤 3 进入 NFS 盘待访问文件所在目录，修改用户自定义配置文件所在父目录的权限为 755。

chmod 755 -R /<文件所在路径的父目录路径>

步骤 4 确认 Core 或者 Task 节点是否可以访问到该配置文件。

1. 以 root 用户登录 Core/Task 节点。
   如果当前集群已启用 Kerberos 认证，请以 root 用户登录 Core 节点。
2. 执行 **su - yarn_user** 命令切换到 yarn_user 用户。
   如果当前集群已启用 Kerberos 认证，请执行 **su - 提交作业的用户**命令切换用户。

3. 执行如下命令查看用户权限，文件所在路径请使用该文件的绝对路径。

   ll <文件所在路径>

----结束

建议与总结

当用户提交的任务中要访问自定义的配置文件时，特别是挂载 NFS 盘时，除了确认文件的权限之外，还要确认文件所在父目录的权限是否正确。NFS 盘挂载到 MRS 集群节点上，一般会新建软连接到 NFS 目录，这个时候需要查看 NFS 上的目录权限是否正确。

6.6 自定义 Flink log4j 日志输出级别

用户问题

MRS 3.1.0 集群自定义 Flink log4j 日志级别不生效。

问题现象

1. 在使用 MRS 3.1.0 集群 Flink 数据分析时，将$Flink_HOME/conf 目录下的 log4j.properties 文件中日志级别修改为 INFO 级别日志。
2. 任务正常提交后，console 未打印出 INFO 级别日志，输出的日志级别还是 ERROR 级别。

原因分析

修改$Flink_HOME/conf 目录下的 log4j.properties 文件，控制的是 JobManager 和 TaskManager 的算子内的日志输出，输出的日志会打印到对应的 yarn contain 中，可以在 yarn web ui 查看对应日志。MRS 3.1.0 及之后版本的 Flink 1.12.0 版本开始默认的日志框架是 log4j2，配置的方式跟之前 log4j 的方式有区别，使用如 log4j 日志规则不会生效。

处理步骤

Log4j2 详细日志规格配置参考开源官方文档：
http://logging.apache.org/log4j/2.x/manual/configuration.html#Properties

7 使用 Flume

7.1 Flume 向 Spark Streaming 提交作业，提交到集群后报类找不到

用户问题

Flume 向 Spark Streaming 提交作业，提交到集群后报类找不到。

问题现象

Spark Streaming 代码打成 jar 包提交到集群后报类找不到错误，通过以下两种方式依然不生效。

1. 在提交 Spark 作业的时候使用**--jars** 命令引用类所在的 jar 包。
2. 将类所在的 jar 包引入 Spark Streaming 的 jar 包。

原因分析

执行 Spark 作业时无法加载部分 jar，导致找不到 class。

处理步骤

步骤 1 使用 **--jars** 加载 flume-ng-sdk-{version}.jar 依赖包。

步骤 2 同时修改 **spark-default.conf** 中两个配置项。

spark.driver.extraClassPath=$PWD/*:{加上原来配置的值}

spark.executor.extraClassPath =$PWD/*

步骤 3 作业运行成功。如果还有报错，则需要排查还有哪个 jar 没有加载，再次执行步骤 1 和步骤 2。

----结束

7.2 Flume 客户端安装失败

问题现象

安装 Flume 客户端失败，提示 JAVA_HOME is null 或 flume has been installed。

```
CST 2016-08-31 17:02:51 [flume-client install]: JAVA HOME is null in current
user,please install the JDK and set the JAVA HOME
CST 2016-08-31 17:02:51 [flume-client install]: check environment failed.
CST 2016-08-31 17:02:51 [flume-client install]: check param failed.
CST 2016-08-31 17:02:51 [flume-client install]: install flume client failed.
CST 2016-08-31 17:03:58 [flume-client install]: flume has been installed
CST 2016-08-31 17:03:58 [flume-client install]: check path failed.
CST 2016-08-31 17:03:58 [flume-client install]: check param failed.
CST 2016-08-31 17:03:58 [flume-client install]: install flume client failed.
```

原因分析

- Flume 客户端安装时会检查环境变量，如果没有可用的 JAVA，会报 JAVA_HOME is null 错误并且退出安装。
- 如果指定的目录下已经安装有 flume，客户端安装时会报 flume has been installed 并退出安装。

解决办法

步骤 1 如果报 JAVA_HOME is null 错误，需要使用命令：

export JAVA_HOME=*java 路径*

设置 JAVA_HOME，重新运行安装脚本。

步骤 2 如果指定的目录下已经安装有 Flume 客户端，需要先卸载已经存在的 Flume 客户端，或指定其他目录安装。

----结束

7.3 Flume 客户端无法连接服务端

问题现象

安装 Flume 客户端并设置 avro sink 与服务端通信，发现无法连接 Flume 服务端。

原因分析

1. 服务端配置错误，监听端口启动失败，例如服务端 avro source 配置了错误的 IP，或者已经被占用了的端口。查看 Flume 运行日志：

```
2016-08-31 17:28:42,092 | ERROR | [lifecycleSupervisor-1-9] |  Unable to start
EventDrivenSourceRunner: { source:Avro source avro_source: { bindAddress:
10.120.205.7, port: 21154 } } - Exception follows.  |
org.apache.flume.lifecycle.LifecycleSupervisor$MonitorRunnable.run(LifecycleSup
```

```
ervisor.java:253)
java.lang.RuntimeException: org.jboss.netty.channel.ChannelException: Failed to
bind to: /192.168.205.7:21154
```

2. 若采用了加密传输，证书或密码错误。

```
2016-08-31 17:15:59,593 | ERROR | [conf-file-poller-0] |  Source avro_source
has been removed due to an error during configuration  |
org.apache.flume.node.AbstractConfigurationProvider.loadSources(AbstractConfigu
rationProvider.java:388)
org.apache.flume.FlumeException: Avro source configured with invalid keystore:
/opt/Bigdata/MRS_XXX/install/FusionInsight-Flume-
1.9.0/flume/conf/flume_sChat.jks
```

3. 客户端与服务端通信异常。

```
PING 192.168.85.55 (10.120.85.55) 56(84) bytes of data.
From 192.168.85.50 icmp_seq=1 Destination Host Unreachable
From 192.168.85.50 icmp_seq=2 Destination Host Unreachable
From 192.168.85.50 icmp_seq=3 Destination Host Unreachable
From 192.168.85.50 icmp_seq=4 Destination Host Unreachable
```

解决办法

步骤 1 设置为正确的 IP，必须为本机的 IP，如果端口被占用，重新配置一个空闲的端口。

步骤 2 配置正确的证书路径。

步骤 3 联系网络管理员，恢复网络。

----结束

7.4 Flume 数据写入组件失败

问题现象

Flume 进程启动后，Flume 数据无法写入到对应组件。（以下以服务端写入到 HDFS 为例）

原因分析

1. HDFS 未启动或故障。查看 Flume 运行日志：

```
2019-02-26 11:16:33,564 | ERROR | [SinkRunner-PollingRunner-
DefaultSinkProcessor] |  opreation the hdfs file errors.  |
org.apache.flume.sink.hdfs.HDFSEventSink.process(HDFSEventSink.java:414)
2019-02-26 11:16:33,747 | WARN  | [hdfs-CCCC-call-runner-4] |  A failover has
occurred since the start of call #32795
ClientNamenodeProtocolTranslatorPB.getFileInfo over 192-168-13-
88/192.168.13.88:25000  |
org.apache.hadoop.io.retry.RetryInvocationHandler$ProxyDescriptor.failover(Retr
yInvocationHandler.java:220)
2019-02-26 11:16:33,748 | ERROR | [hdfs-CCCC-call-runner-4] |  execute hdfs
error. {}  |
org.apache.flume.sink.hdfs.HDFSEventSink$3.call(HDFSEventSink.java:744)
```

```
java.net.ConnectException: Call From 192-168-12-221/192.168.12.221 to 192-168-13-88:25000 failed on connection exception: java.net.ConnectException: Connection refused; For more details see: http://wiki.apache.org/hadoop/ConnectionRefused
```

2. hdfs sink 未启动。查看 Flume 运行日志，发现“ flume current metrics”中并没有 sink 信息：

```
2019-02-26 11:46:05,501 | INFO  | [pool-22-thread-1] |  flume current metrics:{"CHANNEL.BBBB":{"ChannelCapacity":"10000","ChannelFillPercentage":"0.0","Type":"CHANNEL","ChannelStoreSize":"0","EventProcessTimedelta":"0","EventTakeSuccessCount":"0","ChannelSize":"0","EventTakeAttemptCount":"0","StartTime":"1551152734999","EventPutAttemptCount":"0","EventPutSuccessCount":"0","StopTime":"0"},"SOURCE.AAAA":{"AppendBatchAcceptedCount":"0","EventAcceptedCount":"0","AppendReceivedCount":"0","MonTime":"0","StartTime":"1551152735503","AppendBatchReceivedCount":"0","EventReceivedCount":"0","Type":"SOURCE","TotalFilesCount":"1001","SizeAcceptedCount":"0","UpdateTime":"605410241202740","AppendAcceptedCount":"0","OpenConnectionCount":"0","MovedFilesCount":"1001","StopTime":"0"}}  | org.apache.flume.node.Application.getRestartComps(Application.java:467)
```

解决办法

步骤 1 若 Flume 数据写入的组件未启动，启动对应组件；若组件异常，请联系服务技术支持。

步骤 2 sink 未启动，检查配置文件是否配置正确，若配置错误，则正确修改配置文件后重启 Flume 进程，如果配置正确，则查看日志错误信息，根据具体错误信息指定解决办法。

----结束

7.5 Flume 服务端进程故障

问题现象

Flume 运行一段时间后，Manager 界面 Flume 实例显示运行状态“故障”。

原因分析

Flume 文件或文件夹权限异常，重启后 Manager 界面提示如下信息：

```
[2019-02-26 13:38:02]RoleInstance prepare to start failure [{ScriptExecutionResult=ScriptExecutionResult [exitCode=126, output=, errMsg=sh: line 1: /opt/Bigdata/MRS_XXX/install/FusionInsight-Flume-1.9.0/flume/bin/flume-manage.sh: Permission denied
```

解决办法

与运行正常的 Flume 节点进行文件和文件夹权限对比，更改错误文件或文件夹权限。

7.6 Flume 数据采集慢

问题现象

Flume 启动后，Flume 数据采集慢。

原因分析

1. Flume 堆内存设置不合理，导致 Flume 进程一直处于频繁 GC。查看 Flume 运行日志：

```
2019-02-26T13:06:20.666+0800: 1085673.512: [Full GC:[CMS: 3849339k->3843458K(3853568K), 2.5817610 secs] 4153654K->3843458K(4160256K), [CMS Perm : 27335K->27335K(45592K),2.5820080 SECS] [Times: user=2.63, sys0.00, real=2.59 secs]
```

2. 用户业务配置的 Spooldir source 的 deletePolicy 策略是立即删除（immediate）。

解决办法

步骤 1 适当调大堆内存（xmx）的值。

步骤 2 将 Spooldir source 的 deletePolicy 策略更改为永不删除（never）。

----结束

7.7 Flume 启动失败

问题现象

安装 Flume 服务或重启 Flume 服务失败。

原因分析

1. Flume 堆内存设置的值大于机器剩余内存，查看 Flume 启动日志：

```
[CST 2019-02-26 13:31:43][INFO] [[checkMemoryValidity:124]] [GC_OPTS is invalid: Xmx(40960000MB) is bigger than the free memory(56118MB) in system.] [9928]
```

2. Flume 文件或文件夹权限异常，界面或后台会提示如下信息：

```
[2019-02-26 13:38:02]RoleInstance prepare to start failure [{ScriptExecutionResult=ScriptExecutionResult [exitCode=126, output=, errMsg=sh: line 1: /opt/Bigdata/MRS XXX/install/FusionInsight-Flume-1.9.0/flume/bin/flume-manage.sh: Permission denied
```

3. JAVA_HOME 配置错误，查看 Flume agent 启动日志：

```
Info: Sourcing environment configuration script /opt/FlumeClient/fusioninsight-flume-1.9.0/conf/flume-env.sh
+ '[' -n '' ']'
+ exec /tmp/MRS-Client/MRS_Flume_ClientConfig/JDK/jdk-8u18/bin/java '-XX:OnOutOfMemoryError=bash /opt/FlumeClient/fusioninsight-flume-1.9.0/bin/out_memory_error.sh /opt/FlumeClient/fusioninsight-flume-1.9.0/conf %p' -Xms2G -Xmx4G -XX:CMSFullGCsBeforeCompaction=1 -
```

```
XX:+UseConcMarkSweepGC -XX:+CMSParallelRemarkEnabled -
XX:+UseCMSCompactAtFullCollection -Dkerberos.domain.name=hadoop.hadoop.com -
verbose:gc -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -
XX:GCLogFileSize=1M -XX:+PrintGCDetails -XX:+PrintGCDateStamps -
Xloggc:/var/log/Bigdata//flume-client-1/flume/flume-root-20190226134231-%p-
gc.log -Dproc_org.apache.flume.node.Application -Dproc_name=client -
Dproc_conf_file=/opt/FlumeClient/fusioninsight-flume-
1.9.0/conf/properties.properties -
Djava.security.krb5.conf=/opt/FlumeClient/fusioninsight-flume-
1.9.0/conf//krb5.conf -
Djava.security.auth.login.config=/opt/FlumeClient/fusioninsight-flume-
1.9.0/conf//jaas.conf -Dzookeeper.server.principal=zookeeper/hadoop.hadoop.com
-Dzookeeper.request.timeout=120000 -Dflume.instance.id=884174180 -
Dflume.agent.name=clientName1 -Dflume.role=client -
Dlog4j.configuration.watch=true -Dlog4j.configuration=log4j.properties -
Dflume_log_dir=/var/log/Bigdata//flume-client-1/flume/ -
Dflume.service.id=flume-client-1 -
Dbeetle.application.home.path=/opt/FlumeClient/fusioninsight-flume-
1.9.0/conf/service -Dflume.called.from.service -
Dflume.conf.dir=/opt/FlumeClient/fusioninsight-flume-1.9.0/conf -
Dflume.metric.conf.dir=/opt/FlumeClient/fusioninsight-flume-1.9.0/conf -
Dflume.script.home=/opt/FlumeClient/fusioninsight-flume-1.9.0/bin -cp
'/opt/FlumeClient/fusioninsight-flume-
1.9.0/conf:/opt/FlumeClient/fusioninsight-flume-
1.9.0/lib/*:/opt/FlumeClient/fusioninsight-flume-1.9.0/conf/service/' -
Djava.library.path=/opt/FlumeClient/fusioninsight-flume-
1.9.0/plugins.d/native/native org.apache.flume.node.Application --conf-file
/opt/FlumeClient/fusioninsight-flume-1.9.0/conf/properties.properties --name
client
/opt/FlumeClient/fusioninsight-flume-1.9.0/bin/flume-ng: line 233:
/tmp/FusionInsight-Client/Flume/FusionInsight_Flume_ClientConfig/JDK/jdk-
8u18/bin/java: No such file or directory
```

解决办法

步骤 1 适当调大堆内存（xmx）的值。

步骤 2 与正常启动 Flume 的节点进行文件和文件夹权限对比，更改错误文件或文件夹权限。

步骤 3 重新配置 JAVA_HOME。客户端替换${install_home}/fusioninsight-flume-*flume 组件版本号*/conf/ENV_VARS 文件中 JAVA_HOME 的值，服务端替换 etc 目录下 ENV_VARS 文件中 JAVA_HOME 的值。

其中 JAVA_HOME 的值可通过登录正常启动 Flume 的节点，执行 **echo ${JAVA_HOME}** 获取。

说明

${install_home}为 Flume 客户端的安装路径。

----结束

8 使用 HBase

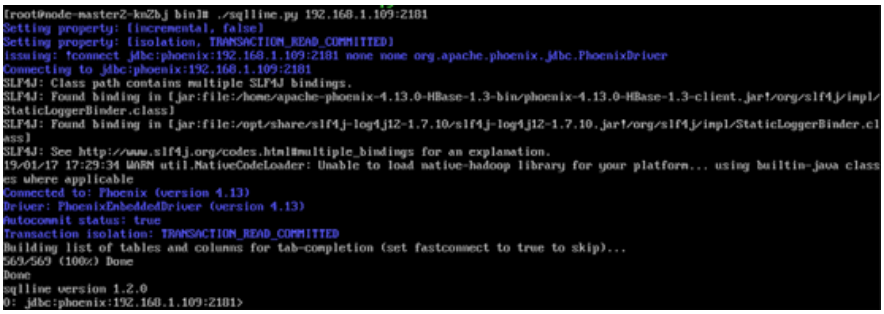
8.1 连接到 HBase 响应慢

用户问题

在相同的 vpc 网络下，外部集群通过 Phoenix 连接到 HBase 响应慢。

问题现象

在相同的 vpc 下，外部集群通过 Phoenix 连接到 HBase 时，响应太慢。



原因分析

客户配置了 DNS 服务，由于客户端连接到 HBase 先通过 DNS 来解析服务器端，所以导致响应慢。

处理步骤

步骤 1 以 root 用户登录 Master 节点。

步骤 2 执行 **vi /etc/resolv.conf**，打开 resolv.conf 文件，注释掉 DNS 服务器地址。例如，#1.1.1.1

----结束

8.2 HBase 用户认证失败

用户问题

HBase 用户认证失败。

问题现象

客户侧 HBase 用户认证失败，报错信息如下：

```
2019-05-13 10:53:09,975 ERROR [localhost-startStop-1] xxxConfig.LoginUtil: login
failed with hbaseuser and /usr/local/linoseyc/hbase-
tomcat/webapps/bigdata_hbase/WEB-INF/classes/user.keytab.
2019-05-13 10:53:09,975 ERROR [localhost-startStop-1] xxxConfig.LoginUtil: perhaps
cause 1 is (wrong password) keytab file and user not match, you can kinit -k -t
keytab user in client server to check.
2019-05-13 10:53:09,975 ERROR [localhost-startStop-1] xxxConfig.LoginUtil: perhaps
cause 2 is (clock skew) time of local server and remote server not match, please
check ntp to remote server.
2019-05-13 10:53:09,975 ERROR [localhost-startStop-1] xxxConfig.LoginUtil: perhaps
cause 3 is (aes256 not support) aes256 not support by default jdk/jre, need copy
local_policy.jar and US_export_policy.jar from remote server in path
${BIGDATA_HOME}/jdk/jre/lib/security.
```

原因分析

客户使用的 JDK 中的 jar 包与 MRS 服务认证的 jar 包版本不一致。

处理步骤

步骤 1　以 root 登录集群 Master1 节点。

步骤 2　执行如下命令，查看 MRS 服务认证的 jar 包。

ll /opt/share/local_policy/local_policy.jar

ll /opt/Bigdata/jdk{version}/jre/lib/security/local_policy.jar

步骤 3　将步骤 2 中的 jar 包下载到本地。

步骤 4　将下载的 jar 包替换到本地 JDK 目录/opt/Bigdata/jdk/jre/lib/security。

步骤 5　执行 **cd** *客户端安装目录*/**HBase/hbase/bin** 命令，进入到 HBase 的 bin 目录。

步骤 6　执行 **sh　start-hbase.sh** 命令，重启 HBase 组件。

----结束

8.3 端口被占用导致 RegionServer 启动失败

问题现象

Manager 页面监控发现 RegionServer 状态为 Restoring。

原因分析

1. 通过查看 RegionServer 日志（/var/log/Bigdata/hbase/rs/hbase-omm-xxx.log）。
2. 使用 **lsof -i:21302**（MRS1.7.X 及以后端口号是 16020）查看到 pid，然后根据 pid 查看到相应的进程，发现 RegionServer 的端口被 DFSZkFailoverController 占用。
3. 查看“/proc/sys/net/ipv4/ip_local_port_range”显示为“9000　65500”，临时端口范围与 MRS 产品端口范围重叠，因为安装时未进行 preinstall 操作。

解决办法

步骤 1　执行 **kill -9** *DFSZkFailoverController 的 pid*，　使得其重启后绑定其它端口，然后重启 Restoring 的 RegionServer。

----结束

8.4 节点剩余内存不足导致 HBase 启动失败

问题现象

HBase 的 RegionServer 服务一直是 Restoring 状态。

原因分析

1. 查看 RegionServer 的日志（“/var/log/Bigdata/hbase/rs/hbase-omm-XXX.out”），发现显示以下打印信息：

   ```
   There is insufficient memory for the Java Runtime Environment to continue.
   ```

2. 使用 **free** 指令查看，该节点确实没有足够内存。

解决办法

步骤 1　现场进行排查内存不足原因，确认是否有某些进程占用过多内存，或者由于服务器自身内存不足。

----结束

8.5 HDFS 性能差导致 HBase 服务不可用告警

问题现象

HBase 组件断断续续上报服务不可用告警。

原因分析

该问题多半为 HDFS 性能较慢，导致健康检查超时，从而导致监控告警。可通过以下方式判断：

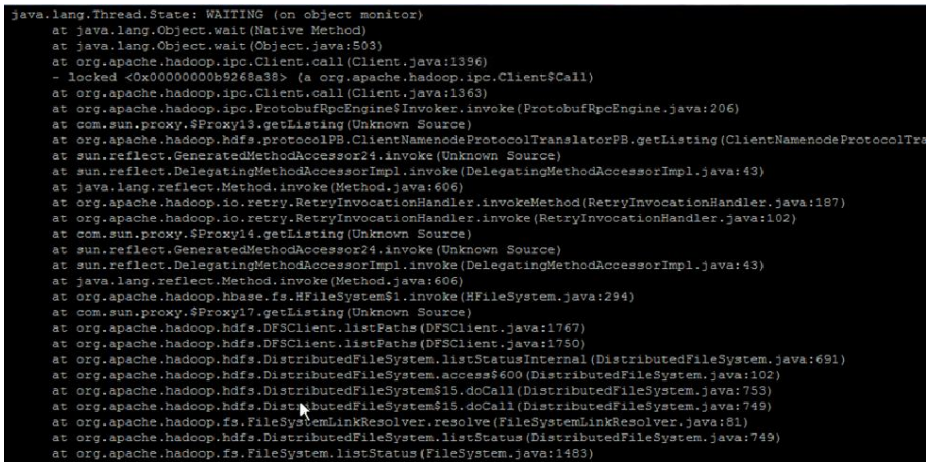
1. 首先查看 HMaster 日志（“/var/log/Bigdata/hbase/hm/hbase-omm-xxx.log”），确认 HMaster 日志中没有频繁打印“system pause”或“jvm”等 GC 相关信息。
2. 然后可以通过下列三种方式确认原因为 HDFS 性能慢造成告警产生。
   a. 使用客户端验证，通过 **hbase shell** 进入 hbase 命令行后，执行 **list** 验证需要运行多久。
   b. 开启 HDFS 的 debug 日志，然后查看下层目录很多的路径（**hadoop fs -ls /XXX/XXX**），验证需要运行多久。
   c. 打印 HMaster 进程 jstack：

      su - omm

      jps

      jstack pid
3. 如下图所示，Jstack 显示一直卡在 DFSClient.listPaths。

图8-1 异常



解决办法

步骤 1 如果确认是 HDFS 性能慢导致告警，需要排除是否为旧版本中 Impala 运行导致 HDFS 性能慢或者是否为集群最初部署时 JournalNode 部署不正确（部署过多，大于 3 个）。

----结束

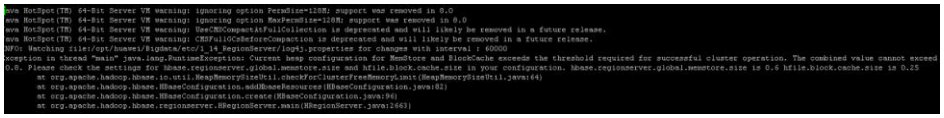
8.6 参数不合理导致 HBase 启动失败

问题现象

修改部分参数后，无法正常启动 HBase。

原因分析

1. 查看 HMaster 日志（/var/log/Bigdata/hbase/hm/hbase-omm-xxx.log）显示，hbase.regionserver.global.memstore.size + hfile.block.cache.size 总和大于 0.8 导致启动不成功，因此需要调整参数配置值总和低于 0.8。



2. 查看 HMaster 和 RegionServer 的 out 日志（/var/log/Bigdata/hbase/hm/hbase-omm-xxx.out/var/log/Bigdata/hbase/rs/hbase-omm-xxx.out），提示 Unrecognized VM option。

```
Unrecognized VM option
Error: Could not create the Java Virtual Machine.
Error: A fatal exception has occurred. Program will exit.
```

检查 GC_OPTS 相关参数存在多余空格，如-D sun.rmi.dgc.server.gcInterval=0x7FFFFFFFFFFFFE。

解决办法

步骤 1 针对 memstore、cache 修改配置参数后，重启 HBase 服务成功。

步骤 2 针对 GC_OPTS 配置错误，修改参数后重启 HBase 服务成功。

----结束

8.7 残留进程导致 Regionsever 启动失败

问题现象

HBase 服务启动失败，健康检查报错。

原因分析

查看启动 HBase 服务时 manager 页面的详细打印信息，提示 the previous process is not quit。

解决办法

步骤 1 登录节点，后台通过执行 **ps -ef | grep HRegionServer** 发现确实存在一个残留的进程。

步骤 2 确认进程可以终止后，使用 kill 命令终止该进程（如果 kill 无法终止该进程，需要使用 kill -9 来强制终止该进程）。

步骤 3 重新启动 HBase 服务成功。

----结束

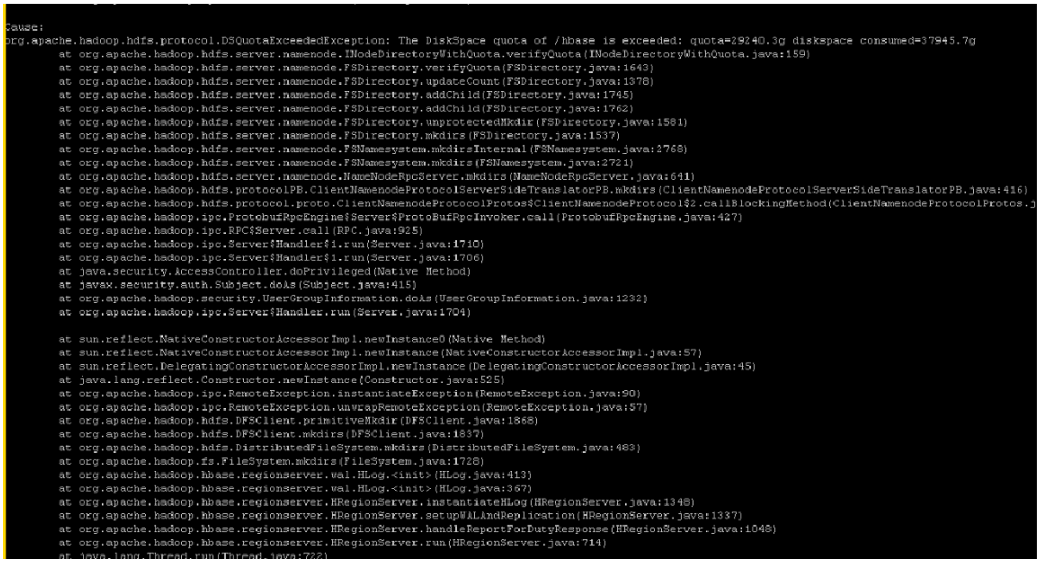
8.8 HDFS 上设置配额导致 HBase 启动失败

问题现象

HBase 启动失败。

原因分析

查看 HMaster 日志信息（“/var/log/Bigdata/hbase/hm/hbase-omm-xxx.log”），出现如下异常，The DiskSpace quota of /hbase is exceeded。



解决办法

步骤 1 通过后台使用 **df -h** 命令查看数据盘目录空间已满，因此需要删除无用的数据来进行应急恢复。

步骤 2 后续需要扩容节点来解决数据目录空间不足问题。

----结束

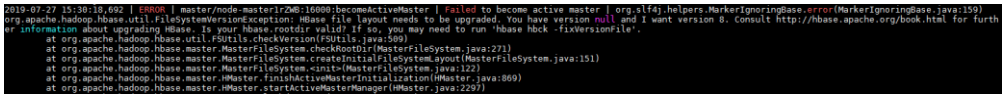
8.9 HBase version 文件损坏导致启动失败

问题背景

HBase 启动失败。

原因分析

1. HBase 启动时会读取 hbase.version 文件，但是日志显示读取存在异常。



2. 通过 **hadoop fs -cat /hbase/hbase.version** 命令发现文件不能正常查看，该文件损坏。

解决办法

步骤 1 执行 **hbase hbck -fixVersionFile** 命令修复文件。

步骤 2 如步骤 1 不能解决，从同版本的其他集群中获取 hbase.version 文件上传进行替换。

步骤 3 重新启动 HBase 服务。

----结束

8.10 无业务情况下，RegionServer 占用 CPU 高

问题背景

无业务情况下，RegionServer 占用 CPU 较高。

原因分析

1. 通过 **top** 命令获取 RegionServer 的进程使用 CPU 情况信息，查看 CPU 使用率高的进程号。
2. 根据 RegionServer 的进程编号，获取该进程下线程使用 CPU 情况。

   top -H -p <PID>（根据实际 RegionServer 的进程 ID 进行替换），具体如下图所示，发现部分线程 CPU 使用率均达到 80%。

```
PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
75706 omm     20   0 6879444   1.0g  25612 S  90.4  1.6   0:00.00 java
75716 omm     20   0 6879444   1.0g  25612 S  90.4  1.6   0:04.74 java
75720 omm     20   0 6879444   1.0g  25612 S  88.6  1.6   0:01.93 java
75721 omm     20   0 6879444   1.0g  25612 S  86.8  1.6   0:01.99 java
75722 omm     20   0 6879444   1.0g  25612 S  86.8  1.6   0:01.94 java
75723 omm     20   0 6879444   1.0g  25612 S  86.8  1.6   0:01.96 java
75724 omm     20   0 6879444   1.0g  25612 S  86.8  1.6   0:01.97 java
75725 omm     20   0 6879444   1.0g  25612 S  81.5  1.6   0:02.06 java
75726 omm     20   0 6879444   1.0g  25612 S  79.7  1.6   0:02.01 java
```

```
75727 omm      20   0 6879444   1.0g  25612 S  79.7  1.6   0:01.95 java
75728 omm      20   0 6879444   1.0g  25612 S  78.0  1.6   0:01.99 java
```

3. 根据 RegionServer 的进程编号，获取线程堆栈信息。

   jstack 12345　>allstack.txt （根据实际 RegionServer 的进程 ID 进行替换）

4. 将需要的线程 ID 转换为 16 进制格式：

   printf "%x\n" 30648

   输出结果 TID 为 77b8。

5. 根据输出 16 进制 TID，在线程堆栈中进行查找，发现在执行 compaction 操作。

```
"regionserver/ahbd-hbase-dat1/12.2.168.1:21302-longCompactions-1482676601478" #1641 prio=5 os_prio=0 tid=0x00007fa614563000 nid=0x77b8 runnable [0x0
   java.lang.Thread.State: RUNNABLE
    at org.apache.hadoop.io.compress.snappy.SnappyCompressor.compressBytesDirect(Native Method)
    at org.apache.hadoop.io.compress.snappy.SnappyCompressor.compress(SnappyCompressor.java:228)
    at org.apache.hadoop.io.compress.BlockCompressorStream.compress(BlockCompressorStream.java:149)
    at org.apache.hadoop.io.compress.BlockCompressorStream.finish(BlockCompressorStream.java:142)
    at org.apache.hadoop.hbase.io.encoding.HFileBlockDefaultEncodingContext.compressAfterEncoding(HFileBlockDefaultEncodingContext.java:219)
    at org.apache.hadoop.hbase.io.encoding.HFileBlockDefaultEncodingContext.compressAndEncrypt(HFileBlockDefaultEncodingContext.java:132)
    at org.apache.hadoop.hbase.io.hfile.HFileBlock$Writer.finishBlock(HFileBlock.java:989)
    at org.apache.hadoop.hbase.io.hfile.HFileBlock$Writer.ensureBlockReady(HFileBlock.java:961)
    at org.apache.hadoop.hbase.io.hfile.HFileBlock$Writer.finishBlockAndWriteHeaderAndData(HFileBlock.java:1077)
```

6. 对其它线程执行相同操作，发现均为 compactions 线程。

```
"regionserver/ahbd-hbase-dat1/12.2.168.1:21302-longCompactions-1482676601473" #1629 prio=5 os_prio=0 tid=0x00007fa61454d800 nid=0x77a0 runnable
   java.lang.Thread.State: RUNNABLE
    at org.apache.hadoop.hdfs.DFSOutputStream.writeChunk(DFSOutputStream.java:425)
    - locked <0x000000020276ba38> (a org.apache.hadoop.hdfs.DFSOutputStream)
    at org.apache.hadoop.fs.FSOutputSummer.writeChecksumChunks(FSOutputSummer.java:214)
    at org.apache.hadoop.fs.FSOutputSummer.flushBuffer(FSOutputSummer.java:165)
    - locked <0x000000020276ba38> (a org.apache.hadoop.hdfs.DFSOutputStream)
    at org.apache.hadoop.fs.FSOutputSummer.flushBuffer(FSOutputSummer.java:146)
    - eliminated <0x000000020276ba38> (a org.apache.hadoop.hdfs.DFSOutputStream)
    at org.apache.hadoop.fs.FSOutputSummer.write1(FSOutputSummer.java:137)
    at org.apache.hadoop.fs.FSOutputSummer.write(FSOutputSummer.java:112)
    - locked <0x000000020276ba38> (a org.apache.hadoop.hdfs.DFSOutputStream)
    at org.apache.hadoop.fs.FSDataOutputStream$PositionCache.write(FSDataOutputStream.java:58)
    at java.io.DataOutputStream.write(DataOutputStream.java:107)
    - locked <0x00000004de9535c8> (a org.apache.hadoop.hdfs.client.HdfsDataOutputStream)
    at java.io.FilterOutputStream.write(FilterOutputStream.java:97)
```

解决办法

属于正常现象。

发现消耗 CPU 较高线程均为 HBase 的 compaction，其中部分线程调用 Snappy 压缩处理，部分线程调用 HDFS 读写数据。当前每个 Region 数据量和数据文件多，且采用 Snappy 压缩算法，因此执行 compaction 时会使用大量 CPU 导致 CPU 较高。

定位办法

步骤 1　使用 **top** 命令查看 CPU 使用率高的进程号。

步骤 2　查看此进程中占用 CPU 高的线程。

使用命令 **top -H -p <PID>**即可打印出某进程<PID>下的线程的 CPU 耗时信息。

一般某个进程如果出现问题，是因为某个线程出现问题了，获取查询到的占用 CPU 最高的线程号。

或者使用命令 **ps -mp <PID> -o THREAD,tid,time | sort -rn**。

观察回显可以得到 CPU 最高的线程号。

步骤 3　获取出现问题的线程的堆栈。

java 问题使用 jstack 工具是最有效，最可靠的。

到 java/bin 目录下有 jstack 工具，获取进程堆栈，并输出到本地文件。

jstack <PID> > allstack.txt

获取线程堆栈，并输出到本地文件。

步骤 4 将需要的线程 ID 转换为 16 进制格式。

printf "%x\n" <PID>

回显结果为线程 ID，即 TID。

步骤 5 使用命令获得 TID,并输出到本地文件。

jstack <PID> | grep <TID> > Onestack.txt

如果只是在命令行窗口查看，可以使用命名：

jstack <PID> | grep <TID> -A 30

-A 30 意思是显示 30 行。

----结束

8.11 HBase 启动失败，RegionServer 日志中提示 FileNotFoundException 异常

问题背景

HBase 启动失败，RegionServer 一直处于 Restoring 状态。

原因分析

1. 查看 RegionServer 的日志（/var/log/Bigdata/hbase/rs/hbase-omm-XXX.log），发现显示以下打印信息：

```
 | ERROR | RS_OPEN_REGION-ab-dn01:21302-2 | ABORTING region server ab-
dn01,21302,1487663269375: The coprocessor
org.apache.kylin.storage.hbase.cube.v2.coprocessor.endpoint.CubeVisitService
threw java.io.FileNotFoundException: File does not exist:
hdfs://hacluster/kylin/kylin_metadata/coprocessor/kylin-coprocessor-1.6.0-
SNAPSHOT-0.jar |
org.apache.hadoop.hbase.regionserver.HRegionServer.abort(HRegionServer.java:212
3)
java.io.FileNotFoundException: File does not exist:
hdfs://hacluster/kylin/kylin_metadata/coprocessor/kylin-coprocessor-1.6.0-
SNAPSHOT-0.jar
at
org.apache.hadoop.hdfs.DistributedFileSystem$25.doCall(DistributedFileSystem.ja
va:1399)
at
org.apache.hadoop.hdfs.DistributedFileSystem$25.doCall(DistributedFileSystem.ja
va:1391)
at
org.apache.hadoop.fs.FileSystemLinkResolver.resolve(FileSystemLinkResolver.java
:81)
```

```
at
org.apache.hadoop.hdfs.DistributedFileSystem.getFileStatus(DistributedFileSyste
m.java:1391)
at org.apache.hadoop.fs.FileUtil.copy(FileUtil.java:340)
at org.apache.hadoop.fs.FileUtil.copy(FileUtil.java:292)
at org.apache.hadoop.fs.FileSystem.copyToLocalFile(FileSystem.java:2038)
at org.apache.hadoop.fs.FileSystem.copyToLocalFile(FileSystem.java:2007)
at org.apache.hadoop.fs.FileSystem.copyToLocalFile(FileSystem.java:1983)
at
org.apache.hadoop.hbase.util.CoprocessorClassLoader.init(CoprocessorClassLoader
.java:168)
at
org.apache.hadoop.hbase.util.CoprocessorClassLoader.getClassLoader(CoprocessorC
lassLoader.java:250)
at
org.apache.hadoop.hbase.coprocessor.CoprocessorHost.load(CoprocessorHost.java:2
24)
at
org.apache.hadoop.hbase.regionserver.RegionCoprocessorHost.loadTableCoprocessor
s(RegionCoprocessorHost.java:365)
at
org.apache.hadoop.hbase.regionserver.RegionCoprocessorHost.<init>(RegionCoproce
ssorHost.java:227)
at org.apache.hadoop.hbase.regionserver.HRegion.<init>(HRegion.java:783)
at org.apache.hadoop.hbase.regionserver.HRegion.<init>(HRegion.java:689)
at sun.reflect.GeneratedConstructorAccessor22.newInstance(Unknown Source)
at
sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructor
AccessorImpl.java:45)
at java.lang.reflect.Constructor.newInstance(Constructor.java:423)
at org.apache.hadoop.hbase.regionserver.HRegion.newHRegion(HRegion.java:6312)
at org.apache.hadoop.hbase.regionserver.HRegion.openHRegion(HRegion.java:6622)
at org.apache.hadoop.hbase.regionserver.HRegion.openHRegion(HRegion.java:6594)
at org.apache.hadoop.hbase.regionserver.HRegion.openHRegion(HRegion.java:6550)
at org.apache.hadoop.hbase.regionserver.HRegion.openHRegion(HRegion.java:6501)
at
org.apache.hadoop.hbase.regionserver.handler.OpenRegionHandler.openRegion(OpenR
egionHandler.java:363)
at
org.apache.hadoop.hbase.regionserver.handler.OpenRegionHandler.process(OpenRegi
onHandler.java:129)
at org.apache.hadoop.hbase.executor.EventHandler.run(EventHandler.java:129)
at
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
at java.lang.Thread.run(Thread.java:745)
```

2. 使用客户端 hdfs 命令查看，如下文件不存在。

   hdfs://hacluster/kylin/kylin_metadata/coprocessor/kylin-coprocessor-1.6.0-SNAPSHOT-0.jar

3. HBase 在配置协处理器时，一定要保证对应的 jar 包路径没有问题，否则 HBase 会无法启动。

解决办法

使用 Kylin 对接 MRS，确保 Kylin 相关 jar 包存在。

8.12 HBase 启动后原生页面显示 RegionServer 个数多于实际个数

问题背景

HBase 启动后，HMaster 原生页面显示 RegionServer 个数多于实际 RegionServer 个数。

查看 HMaster 原生页面，显示有 4 个 RegionServer 在线，如下图示：

ServerName	Start time	Requests Per Second	Num. Regions
controller-192-168-1-1,21302,1494933959261	Tue May 16 19:25:59 CST 2017	0	19
controller-192-168-1-2,21302,1494933957536	Tue May 16 19:25:57 CST 2017	0	24
controller-192-168-1-3,21302,1494933958592	Tue May 16 19:25:58 CST 2017	0	16
eth0,21302,1494933958592	Tue May 16 19:25:58 CST 2017	0	0
Total:4		0	59

原因分析

如下图可以看出，第三行 hostname 为 controller-192-168-1-3 节点和第四行 hostname 为 eth0 节点为同一 RegionServer 上报的信息，登录相应节点，查看/etc/hosts 文件，发现，对应同一 ip，配置两个 hostname。如下：

```
# special IPv6 addresses
::1             localhost ipv6-localhost ipv6-loopback

fe00::0         ipv6-localnet

ff00::0         ipv6-mcastprefix
ff02::1         ipv6-allnodes
ff02::2         ipv6-allrouters
ff02::3         ipv6-allhosts
11.1.1.3        eth2 eth2
#192.168.1.3     eth0 eth0
192.168.2.3     eth1 eth1
10.130.87.37    eth3 eth3
192.168.1.102 controller
1.1.1.1 hadoop.hadoop.com
192.168.1.2 controller-192-168-1-2
192.168.1.1 controller-192-168-1-1
192.168.1.3 controller-192-168-1-3
```

解决办法

登录 RegionServer 所在节点，修改/etc/hosts 文件，同一 ip 只能对应同一 hostname。

8.13 RegionServer 实例异常，处于 Restoring 状态

问题背景

HBase 启动失败，RegionServer 一直处于 Restoring 状态。

原因分析

查看异常的 RegionServer 实例的运行日志（/var/log/Bigdata/hbase/rs/hbase-omm-XXX.log），发现显示以下打印信息 ClockOutOfSyncException...，Reported time is too far out of sync with master

```
2017-09-18 11:16:23,636 | FATAL | regionserver21302 | Master rejected startup
because clock is out of sync |
org.apache.hadoop.hbase.regionserver.HRegionServer.reportForDuty(HRegionServer.java
:2059)
org.apache.hadoop.hbase.ClockOutOfSyncException:
org.apache.hadoop.hbase.ClockOutOfSyncException: Server nl-bi-fi-datanode-24-
65,21302,1505726180086 has been rejected; Reported time is too far out of sync with
master.  Time difference of 152109ms > max allowed of 30000ms
at
org.apache.hadoop.hbase.master.ServerManager.checkClockSkew(ServerManager.java:354)
...

...
2017-09-18 11:16:23,858 | ERROR | main | Region server exiting |
org.apache.hadoop.hbase.regionserver.HRegionServerCommandLine.start(HRegionServerCo
mmandLine.java:70)
java.lang.RuntimeException: HRegionServer Aborted
```

该日志说明异常的 RegionServer 实例和 HMaster 实例的时差大于允许的时差值 30s（由参数 hbase.regionserver.maxclockskew 控制，默认 30000ms），导致 RegionServer 实例异常。

解决办法

调整异常节点时间，确保节点间时差小于 30s。

8.14 新安装的集群 HBase 启动失败

问题背景

新安装的集群 HBase 启动失败，查看 RegionServer 日志报如下错误：

```
2018-02-24 16:53:03,863 | ERROR | regionserver/host3/187.6.71.69:21302 | Master
passed us a different hostname to use; was=host3, but now=187-6-71-69 |
org.apache.hadoop.hbase.regionserver.HRegionServer.handleReportForDutyResponse(HReg
ionServer.java:1386)
```

原因分析

/etc/hosts 中同一个 IP 地址配置了多个主机名映射关系。

解决办法

步骤 1　修改/etc/host 中 IP 与主机名的映射关系，配置正确。

步骤 2　重新启动 HBase 组件。

----结束

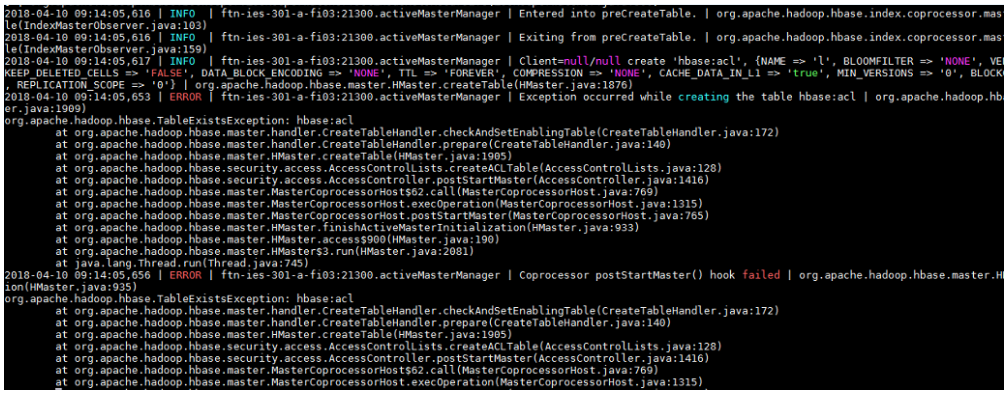
8.15 acl 表目录丢失导致 HBase 启动失败

问题背景与现象

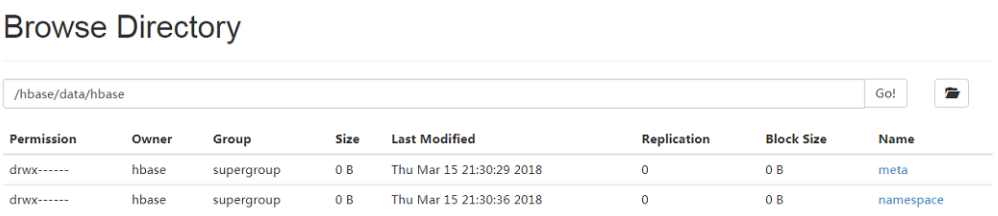
集群 HBase 启动失败

原因分析

1. 查看 HBase 的 HMaster 日志，报如下错误：



2. 检查 HDFS 上 HBase 的路径发现 acl 表路径丢失。



解决办法

步骤 1　停止 HBase 组件。

步骤 2 在 HBase 客户端使用 hbase 用户登录认证，执行如下命令。

例如：

```
hadoop03:~ # source /opt/client/bigdata_env
hadoop03:~ # kinit hbase
Password for hbase@HADOOP.COM:
hadoop03:~ # hbase zkcli
```

步骤 3 删除 zk 中 acl 表信息。

例如：

```
[zk: hadoop01:24002,hadoop02:24002,hadoop03:24002(CONNECTED) 0] deleteall
/hbase/table/hbase:acl
[zk: hadoop01:24002,hadoop02:24002,hadoop03:24002(CONNECTED) 0] deleteall
/hbase/table-lock/hbase:acl
```

步骤 4 启动 HBase 组件。

----结束

8.16 集群上下电之后 HBase 启动失败

问题背景与现象

集群的 ECS 关机重启后，HBase 启动失败。

原因分析

查看 HMaster 的运行日志，发现有报大量的如下错误：

```
2018-03-26 11:10:54,185 | INFO  |
hadoopc1h3,21300,1522031630949_splitLogManager__ChoreService_1 | total tasks = 1
unassigned = 0 tasks={/hbase/splitWAL/WALs%2Fhadoopc1h1%2C213
02%2C1520214023667-
splitting%2Fhadoopc1h1%252C21302%252C1520214023667.default.1520584926990=last_updat
e = 1522033841041 last_version = 34255 cur_worker_name = hadoopc1h3,21302,
1520943011826 status = in_progress incarnation = 3 resubmits = 3 batch = installed
= 1 done = 0 error = 0} |
org.apache.hadoop.hbase.master.SplitLogManager$TimeoutMonitor.chore
(SplitLogManager.java:745)
2018-03-26 11:11:00,185 | INFO  |
hadoopc1h3,21300,1522031630949_splitLogManager__ChoreService_1 | total tasks = 1
unassigned = 0 tasks={/hbase/splitWAL/WALs%2Fhadoopc1h1%2C213
02%2C1520214023667-
splitting%2Fhadoopc1h1%252C21302%252C1520214023667.default.1520584926990=last_updat
e = 1522033841041 last_version = 34255 cur_worker_name = hadoopc1h3,21302,
1520943011826 status = in_progress incarnation = 3 resubmits = 3 batch = installed
= 1 done = 0 error = 0} |
org.apache.hadoop.hbase.master.SplitLogManager$TimeoutMonitor.chore
(SplitLogManager.java:745)
2018-03-26 11:11:06,185 | INFO  |
hadoopc1h3,21300,1522031630949_splitLogManager__ChoreService_1 | total tasks = 1
```

```
unassigned = 0 tasks={/hbase/splitWAL/WALs%2Fhadoopc1h1%2C213
02%2C1520214023667-
splitting%2Fhadoopc1h1%252C21302%252C1520214023667.default.1520584926990=last_updat
e = 1522033841041 last_version = 34255 cur_worker_name = hadoopc1h3,21302,
1520943011826 status = in_progress incarnation = 3 resubmits = 3 batch = installed
= 1 done = 0 error = 0} |
org.apache.hadoop.hbase.master.SplitLogManager$TimeoutMonitor.chore
(SplitLogManager.java:745)
2018-03-26 11:11:10,787 | INFO  |
RpcServer.reader=9,bindAddress=hadoopc1h3,port=21300 | Kerberos principal name is
hbase/hadoop.hadoop.com@HADOOP.COM | org.apache.hadoop.hbase
.ipc.RpcServer$Connection.readPreamble(RpcServer.java:1532)
2018-03-26 11:11:12,185 | INFO  |
hadoopc1h3,21300,1522031630949_splitLogManager__ChoreService_1 | total tasks = 1
unassigned = 0 tasks={/hbase/splitWAL/WALs%2Fhadoopc1h1%2C213
02%2C1520214023667-
splitting%2Fhadoopc1h1%252C21302%252C1520214023667.default.1520584926990=last_updat
e = 1522033841041 last_version = 34255 cur_worker_name = hadoopc1h3,21302,
1520943011826 status = in_progress incarnation = 3 resubmits = 3 batch = installed
= 1 done = 0 error = 0} |
org.apache.hadoop.hbase.master.SplitLogManager$TimeoutMonitor.chore
(SplitLogManager.java:745)
2018-03-26 11:11:18,185 | INFO  |
hadoopc1h3,21300,1522031630949 splitLogManager  ChoreService 1 | total tasks = 1
unassigned = 0 tasks={/hbase/splitWAL/WALs%2Fhadoopc1h1%2C213
02%2C1520214023667-
splitting%2Fhadoopc1h1%252C21302%252C1520214023667.default.1520584926990=last updat
e = 1522033841041 last version = 34255 cur worker name = hadoopc1h3,21302,
1520943011826 status = in_progress incarnation = 3 resubmits = 3 batch = installed
= 1 done = 0 error = 0} |
org.apache.hadoop.hbase.master.SplitLogManager$TimeoutMonitor.chore
(SplitLogManager.java:745)
```

节点上下电，RegionServer 的 wal 分裂失败导致。

解决办法

步骤 1　停止 HBase 组件。

步骤 2　通过 hdfs　fsck 命令检查/hbase/WALs 文件的健康状态。

hdfs fsck　/hbase/WALs

输出如下表示文件都正常，如果有异常则需要先处理异常的文件，再执行后面的操作。

```
The filesystem under path '/hbase/WALs' is HEALTHY
```

步骤 3　备份/hbase/WALs 文件。

hdfs dfs -mv /hbase/WALs /hbase/WALs_old

步骤 4　新建/hbase/WALs 目录。

hdfs dfs -mkdir /hbase/WALs

必须保证路径权限是 hbase:hadoop。

步骤 5 启动 HBase 组件。

----结束

8.17 文件块过大导致 HBase 数据导入失败

问题现象

导入数据到 hbase 报错：NotServingRegionException。

原因分析

当一个 block size 大于 2G 时，hdfs 在 seek 的时候会出现读取异常，持续频繁写入 regionserver 时出现了 full gc，且时间比较久，导致 hmaster 与 regionserver 之间的心跳异常，然后 hmaster 把 regionserver 标记为 dead 状态，强制重启了 Regionserver，重启后触发 servercrash 机制开始回滚 wal 日志。现在这个 splitwal 的文件已经达到将近 2.1G，且其仅有一个 block 块，导致 hdfs seek 异常，引起 splitwal 失败，regionserver 检测到当前这个 wal 日志还需要 split，又会触发 splitwal 日志的机制进行回滚，就这样在 split 与 split 失败之间不停循环，导致无法上线该 regionserver 节点上的 region，最后出现查询该 RS 上某一个 region 时会报 region not online 的异常。

处理步骤

步骤 1 在“HMaster Web UI”右侧，单击“HMaster (主)”进入 HBase Web UI 界面。

步骤 2 在“Procedures”页签查看问题节点。

步骤 3 以 root 用户登录问题节并执行 **hdfs dfs -ls** 命令查看所有块信息。

步骤 4 执行 **hdfs dfs -mkdir** 命令新建目录用于存放问题块。

步骤 5 执行 **hdfs dfs -mv** 将问题块转移至新建目录位置。

----结束

建议与总结

以下两点可供参考：

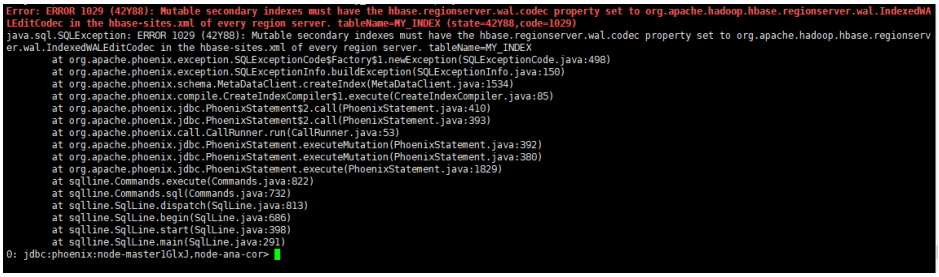
- 数据块损坏，通过 **hdfs fsck /tmp -files -blocks -racks** 命令检查 block 数据块健康信息。
- region 正在分裂时对数据的操作会抛 NotServingRegionException 异常。

8.18 使用 Phoenix 创建 HBase 表后，向索引表中加载数据报错

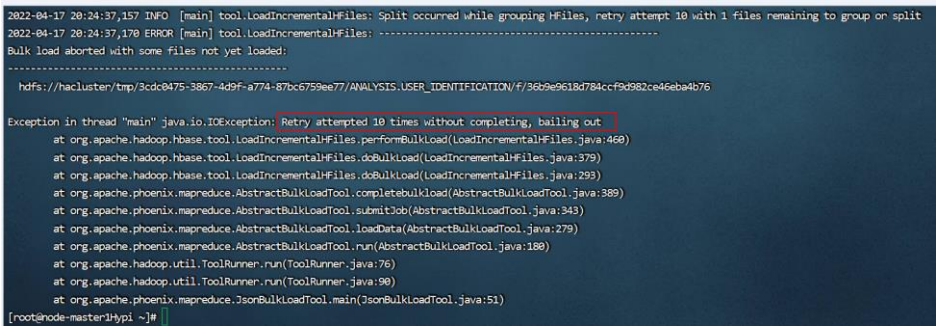
问题背景与现象

使用 Phoenix 创建 HBase 表后，使用命令向索引表中加载数据报错：

- MRS 2.x 及之前版本：Mutable secondary indexes must have the hbase.regionserver.wal.codec property set to org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec in the hbase-sites.xml of every region server. tableName=MY_INDEX (state=42Y88,code=1029)



- MRS 3.x 及之后版本：Exception in thread "main" java.io.IOException: Retry attempted 10 times without completing, bailing out



处理步骤

步骤 1 MRS 2.x 及之前版本，操作步骤如下：

1. 使用 **admin** 用户登录 MRS Manager 界面，选择“服务管理 > HBase > 服务配置”，将“参数类别”的“基础配置”切换为“全部配置”，选择“HMaster > 自定义”，给参数“hbase.hmaster.config.expandor”新增名称为“hbase.regionserver.wal.codec”，值为“org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec”的配置项。
2. 选择“RegionServer > 自定义”，给参数“hbase.regionserver.config.expandor”新增名称为“hbase.regionserver.wal.codec”，值为“org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec”的配置项，单击“保存配置”，输入当前用户密码，单击“确定”，保存配置。
3. 单击“服务状态”，选择“更多 > 重启服务”，输入当前用户密码，单击“确定”，重启 HBase 服务。

步骤 2 MRS 3.x 及之后版本，操作步骤如下：

1. 使用 **admin** 用户登录 FusionInsight Manager，选择“集群 > 服务 > HBase > 配置 > 全部配置 > RegionServer > 自定义”，给参数“hbase.regionserver.config.expandor”新增名称为“hbase.regionserver.wal.codec”，值为“org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec”的配置项。
2. 选择“HMaster > 自定义”，给参数“hbase.hmaster.config.expandor”新增名称为“hbase.regionserver.wal.codec”，值为“org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec”的配置项。
3. 单击“保存”，在弹出的对话框中单击“确定”，保存配置。
4. 单击“概览”，选择“更多 > 重启服务”，输入当前用户密码，单击“确定”，重启 HBase 服务。

----结束

8.19 在 MRS 集群客户端无法执行 hbase shell 命令

用户问题

在 MRS 集群客户端无法执行 **hbase shell** 命令。

原因分析

- 执行 **hbase shell** 命令前未配置环境变量。
- 当前 MRS 集群未安装 HBase 客户端。

处理步骤

步骤 1 使用 **root** 用户登录安装客户端的节点，切换到客户端安装目录，查看是否安装了 HBase 客户端。

- 是，执行步骤 2。
- 否，下载并安装 HBase 客户端。

步骤 2 执行以下命令配置环境变量。

source bigdata_env

步骤 3 如果当前集群已启用 Kerberos 认证，执行以下命令认证当前用户，当前用户需要具有创建 HBase 表的权限。如果当前集群未启用 Kerberos 认证，则无需执行此命令。

kinit *MRS 集群用户*

步骤 4 执行 HBase 组件的客户端命令。

hbase shell

----结束

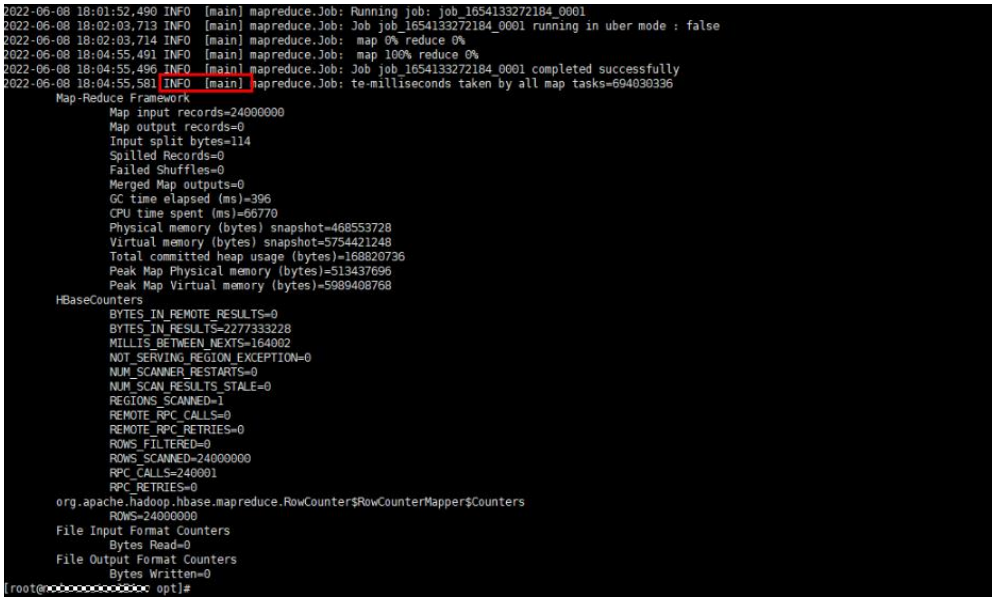
8.20 HBase shell 客户端在使用中有 INFO 信息打印在控制台导致显示混乱

用户问题

在使用 HBase shell 中会有 INFO 信息打印在控制台，导致显示混乱，用户在输入命令前需要按回车符清理控制台。如下图所示：

原因分析

- HBase 客户端中默认日志打印设置为“INFO,console”，所以在使用期间会有 INFO 日志输出到控制台，影响 HBase shell 窗口的显示。
- HBase 客户端命令繁多，例如：**hbase shell**、**hbase hbck**、**hbase org.apache.hadoop.hbase.mapreduce.RowCounter** 等，且后续还会增加。部分命令的输出为 INFO 打印，如果直接把 INFO 关闭会导致部分命令输出结果丢失。例如：RowCounter 输出结果为 INFO 类型：



处理步骤

步骤 1　使用 **root** 用户登录安装 HBase 客户端的节点。

步骤 2　在“*HBase 客户端安装目录*/HBase/component_env”文件中添加如下信息：

export HBASE_ROOT_LOGGER=INFO,RFA

把日志输出到日志文件中，后期如果使用 **hbase org.apache.hadoop.hbase.mapreduce.RowCounter** 等命令，执行结果请在日志文件“*HBase 客户端安装目录*/HBase/hbase/logs/hbase.log”中查看。

步骤 3　切换到 HBase 客户端安装目录，执行以下命令使配置生效。

cd *HBase 客户端安装目录*

source HBase/component_env

----结束

8.21 RegionServer 剩余内存不足导致 HBase 服务启动失败

用户问题

RegionServer 剩余内存不足导致 HBase 服务启动失败。

原因分析

RegionServer 启动时节点剩余内存不足，导致无法启动实例。排查步骤如下：

1. 登录 Master 节点，到“/var/log/Bigdata”查找 HBase 相关日志，HMaster 的日志中报错“connect regionserver timeout ”。
2. 登录到 1 中 HMaster 连接不上的 RegionServer 节点，到“/var/log/Bigdata”查找 HBase 相关日志，RegionServer 报错“error=’Cannot allocate memory’ (errno=12)”。
3. 根据 2 报错判断由于 RegionServer 内存不足导致 RegionServer 启动失败。

处理步骤

步骤 1 登录报错的 RegionServer 节点，执行以下命令查看节点剩余内存：

free -g

步骤 2 执行 **top** 命令查看节点内存使用情况。

步骤 3 根据 **top** 提示结束内存占用多的进程（内存占用多并且非 MRS 自身组件的进程），并重新启动 HBase 服务。

说明

集群的 Core 节点除了 MRS 组件运行占用外，Yarn 上的作业还会被分配到节点运行，占用节点内存。若是由于 Yarn 作业占用内存多导致组件无法正常启动时，建议扩容 Core 节点。

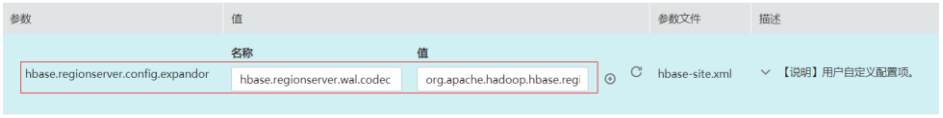
----结束

8.22 集群扩容之后新节点 HRegionServer 启动失败

问题现象

- 集群扩容完成之后，新节点的 HRegionserver 启动失败，一直处于异常状态，无法正常提供服务。

- 登录故障 RegionServer 所在节点，**jps** 查看 RegionServer 进程没有启动，在 Manager 上手动重启故障的 RegionServer 实例失败，查看对应 RegionServer 节点 /var/log/Bigdata/hbase/rs/hbase-omm-regionserver-node-ana-coreqRvt.log 日志，有报错信息“ClassNotFound：org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec”。
- Manager 上，可以看到 RegionServer 有如下自定义配置：



原因分析

用户配置了 Phoenix 的索引功能，由于新节点没有 Phoenix 对应的 jar 包，导致找不到类，启动失败。

处理步骤

步骤 1 登录到正常的 RegionServer 节点，执行以下命令。

grep -Rn ‘org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec’ /opt/Bigdata/MRS_Current/1_16_RegionServer/

查询出/opt/Bigdata/MRS_Current/1_16_RegionServer/install/hbase/lib 目录下面有两个 Phoenix 包（phoenix-4.14.1-server.jar 和 phoenix-core-4.14.1.jar）包含这个类。

步骤 2 使用 **scp** 命令将上述两个包拷贝到异常 RegionServer 节点上相同目录下，重启故障的 RegionServer，恢复正常。

----结束

9 使用 HDFS

9.1 修改集群 HDFS 服务的 NameNode RPC 端口后，NameNode 都变为备状态

用户问题

通过页面更改 NameNode 的 RPC 端口，随后重启 HDFS 服务，出现所有 NameNode 一直是备状态，导致集群异常。

问题现象

所有 NameNode 都是备状态，导致集群异常。

原因分析

集群安装启动后，如果修改 NameNode 的 RPC 端口，则需要重新格式化 Zkfc 服务来更新 zookeeper 上的节点信息。

处理步骤

步骤 1 登录 Manager，停止 HDFS 服务。

说明

在停止 HDFS 时，建议不要停止相关服务。

步骤 2 停止成功后，登录到被修改了 RPC 端口的 Master 节点。

说明

如果两个 Master 节点都被修改了 RPC 端口，则只需登录其中一个修改即可。

步骤 3 执行 **su - omm** 命令切换到 omm 用户。

说明

如果是安全集群，需要执行 **kinit hdfs** 命令进行认证。

步骤 4 执行如下命令，将环境变量脚本加载到环境中。

cd ${BIGDATA_HOME}/*MRS_X.X.X*/*1_8*_Zkfc/etc

source ${BIGDATA_HOME}/*MRS_X.X.X*/install/FusionInsight-Hadoop-3.1.1/hadoop/sbin/exportENV_VARS.sh

说明

命令中的“*MRS_X.X.X*”和“1_8”根据实际版本而定。

步骤 5 加载完成后，执行如下命令，格式化 Zkfc。

cd ${HADOOP_HOME}/bin

./hdfs zkfc -formatZK

步骤 6 格式化成功后，在 Manager 页面“重启”HDFS 服务。

说明

如果更改了 NameNode 的 RPC 端口，则之前安装的所有客户端都需要刷新配置文件。

----结束

9.2 通过公网 IP 连接主机，使用 HDFS 客户端报错

用户问题

通过公网 IP 连接主机，不能使用 HDFS 客户端，运行 HDFS 提示**-bash: hdfs: command not found**。

问题现象

通过公网 IP 连接主机，不能使用 HDFS 客户端，运行 HDFS 提示**-bash: hdfs: command not found**。

原因分析

用户登录 Master 节点执行命令之前，未设置环境变量。

处理步骤

步骤 1 以 **root** 用户登录任意一个 Master 节点。

步骤 2 执行 **source** *客户端安装目录*/**bigdata_env** 命令，设置环境变量。

步骤 3 执行 **hdfs** 命令即可成功使用 HDFS 客户端。

----结束

9.3 使用 Python 远程连接 HDFS 的端口失败

用户问题

使用 Python 远程连接 HDFS 的端口失败，如何解决？

问题现象

用户使用 Python 远程连接 HDFS 的 50070 端口失败。

原因分析

HDFS 开源 3.0.0 以下版本的默认端口为 50070，3.0.0 及以上的默认端口为 9870。用户使用的端口和 HDFS 版本不匹配导致连接端口失败。

步骤 1 登录集群的主 Master 节点。

步骤 2 执行 **su - omm** 命令，切换到 omm 用户。

步骤 3 执行**/opt/Bigdata/om-0.0.1/sbin/queryVersion.sh** 命令，查看集群中的 HDFS 版本号。

根据版本号确认开源组件的端口号。

步骤 4 执行 **netstat -anp|grep ${port}**命令，查看组件的默认端口号是否存在。

如果不存在，说明用户修改了默认的端口号。请修改为默认端口，再重新连接 HDFS。

如果存在，请联系技术服务。

说明

- ${ port }：表示与组件版本相对应的组件默认端口号。
- 如果用户修改了默认端口号，请使用修改后的端口号连接 HDFS。不建议修改默认端口号。

----结束

9.4 HDFS 容量使用达到 100%，导致上层服务 HBase、Spark 等上报服务不可用

用户问题

集群的 HDFS 容量使用达到 100%，HDFS 服务状态为只读，导致上层服务 HBase、Spark 等上报服务不可用告警。

问题现象

HDFS 使用容量 100%，磁盘容量只使用 85%左右，HDFS 服务状态为只读，导致上层服务 HBase、Spark 等上报服务不可用。

原因分析

当前 NodeManager 和 DataNode 共数据盘使用，MRS 默认预留 15%的数据磁盘空间给非 HDFS 使用，可通过 HDFS 参数 **dfs.datanode.du.reserved.percentage** 修改百分比来控制具体的磁盘占比。

当 HDFS 磁盘使用 100%之后，可通过降低 **dfs.datanode.du.reserved.percentage** 百分比来恢复业务，再进行磁盘扩容。

处理步骤

步骤 1 登录集群任意 Master 节点。

步骤 2 执行 **source /opt/client/bigdata_env** 命令，初始化环境变量。

说明

如果是安全集群，则需要执行 kinit -kt <keytab file> <pricipal name> 进行认证。

步骤 3 执行 **hdfs dfs –put ./startDetail.log /tmp** 命令，测试 HDFS 写文件失败。

```
19/05/12 10:07:32 WARN hdfs.DataStreamer: DataStreamer Exception
org.apache.hadoop.ipc.RemoteException(java.io.IOException): File
/tmp/startDetail.log._COPYING_ could only be replicated to 0 nodes instead of
minReplication (=1). There are 3 datanode(s) running and no node(s) are excluded
in this operation.
```

步骤 4 执行 **hdfs dfsadmin -report** 命令，检查 HDFS 使用容量，发现已经达到 100%。

```
Configured Capacity: 5389790579100 (4.90 TB)
Present Capacity: 5067618628404 (4.61 TB)
DFS Remaining: 133350196 (127.17 MB)
DFS Used: 5067485278208 (4.61 TB)
DFS Used%: 100.00%
Under replicated blocks: 10
Blocks with corrupt replicas: 0
Missing blocks: 0
Missing blocks (with replication factor 1): 0
Pending deletion blocks: 0
```

步骤 5 当 HDFS 使用容量达到 100%时，通过 HDFS 参数 **dfs.datanode.du.reserved.percentage** 修改百分比来控制具体的磁盘占比。

1. 登录 Manager 进入服务配置页面。
   - MRS Manager 界面操作入口：登录 MRS Manager，依次选择 “服务管理 > HDFS> 配置”。
   - FusionInsight Manager 界面操作入口：登录 FusionInsight Manager，选择“集群 > 待操作集群的名称 > 服务 >HDFS > 配置”。
2. 选择“全部配置”，在搜索框中搜索 **dfs.datanode.du.reserved.percentage**。
3. 修改此参数的取值为“10”。

步骤 6 修改完成后，扩容 Core 节点的磁盘个数。。

----结束

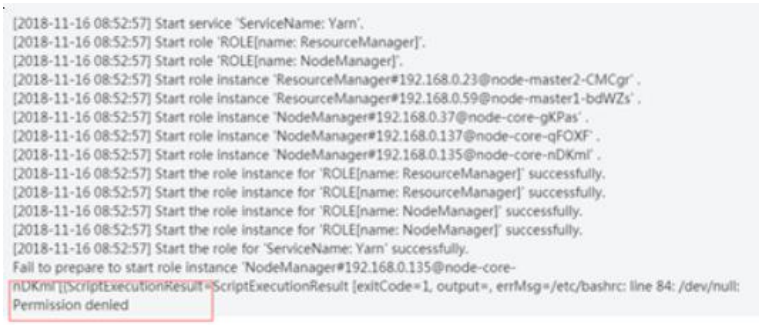
9.5 启动 HDFS 和 Yarn 报错

用户问题

启动 HDFS 和 Yarn 时报错。

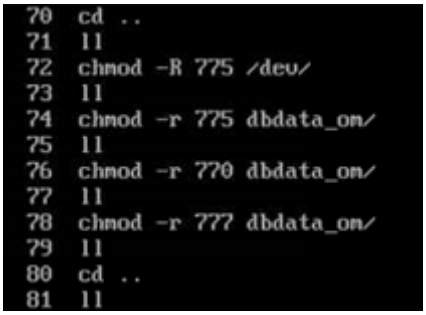
问题现象

无法启动 HDFS、Yarn 服务组件，报错内容：/dev/null Permission denied。



原因分析

客户修改了虚机系统的/dev/null 的权限值为 775。



处理步骤

步骤 1 以 root 用户登录集群的任意一个 Master 节点。

步骤 2 登录成功后，执行 **chmod 666 /dev/null** 命令，修改/dev/null 的权限值为 666。

步骤 3 执行 **ls -al /dev/null** 命令，查看修改的/dev/null 权限值是否为 666，如果不是，需要修改为 666。

步骤 4 修改成功后，重新启动 HDFS 和 Yarn 组件。

----结束

9.6 HDFS 权限设置问题

用户问题

在使用 MRS 服务的时候，某个用户可以在其他用户的 HDFS 目录下面删除或者创建文件。

问题现象

在使用 MRS 服务时，某个用户可以在其他用户的 HDFS 目录下面删除或者创建文件。

原因分析

客户配置的用户具有 ficommon 组的权限，所以可以对 HDFS 任意操作。需要移除用户的 ficommon 组权限。

处理步骤

步骤 1 以 root 用户登录集群的 Master 节点。

步骤 2 执行 **id ${用户名}**命令，显示用户组信息，确认是否有 ficommon 组权限。

如果存在 ficommon 组权限，请执行步骤 3。如果不存在，请联系技术服务。

说明

${用户名}：出现 HDFS 权限设置问题的用户名。

步骤 3 执行 **gpasswd -d ${用户名} ficommon** 命令，删除该用户的 ficommon 组权限。

说明

${用户名}：出现 HDFS 权限设置问题的用户名。

步骤 4 执行成功后，登录 Manager 修改参数。

MRS Manager 界面操作（适用 MRS 3.x 之前版本）：

1. 登录 MRS Manager，在 MRS Manager 页面，选择“服务管理 > HDFS > 服务配置”。
2. “参数类别”选择“全部配置”，在搜索框中输入“dfs.permissions.enabled”，修改为“true”。
3. 修改完成后，单击“保存配置”，并重启 HDFS 服务。

FusionInsight Manager 界面操作（适用 MRS 3.x 及之后版本）：

1. 登录 FusionInsight Manager。选择“集群 > 服务 > HDFS > 配置 > 全部配置”。
2. 在搜索框中输入“dfs.permissions.enabled”，修改为“true”。
3. 修改完成后，单击“保存”，并重启 HDFS 服务。

MRS 集群详情页操作：

1. 登录 MRS 控制台，选择“组件管理 > HDFS > 服务配置”。

2. “参数类别”选择“全部配置”，在搜索框中输入“dfs.permissions.enabled”，修改为“true”。
3. 修改完成后，单击“保存配置”，并重启 HDFS 服务。

----结束

9.7 HDFS 的 DataNode 一直显示退服中

用户问题

HDFS 的 DataNode 一直显示退服中。

问题现象

HDFS 的某个 DataNode 退服（或者对 Core 节点进行缩容）任务失败，但是 DataNode 在任务失败后一直处于退服中的状态。

原因分析

在对 HDFS 的某个 DataNode 进行退服（或者对 core 节点进行缩容）过程中，因为 Master 节点重启或者 nodeagent 进程意外退出等情况出现，使得退服（或缩容）任务失败，并且没有进行黑名单清理。此时 DataNode 节点会一直处于退服中的状态，需要人工介入进行黑名单清理。

处理步骤

步骤 1 进入服务实例界面。

MRS Manager 界面操作：

登录 MRS Manager，在 MRS Manager 页面，选择“服务管理 > HDFS > 实例”。

FusionInsight Manager 界面操作：

对于 MRS 3.x 及后续版本集群：也可登录 FusionInsight Manager。选择“集群 > 服务 > HDFS > 实例”。

也可登录 MRS 控制台，选择“组件管理 > HDFS > 实例”。

步骤 2 查看 HDFS 服务实例状态，找到一直处于退服中的 DataNode，复制这个 DataNode 的 IP 地址。

步骤 3 登录 Master1 节点的后台，执行 **cd ${BIGDATA_HOME}/MRS_*/1_*_NameNode/etc/** 命令进入黑名单目录。

步骤 4 执行 **sed -i "/^*IP*$/d" excludeHosts** 命令清理黑名单中的故障 DataNode 信息，该命令中 IP 替换为步骤 2 中查询到的故障 DataNode 的 IP 地址，其中不能有空格。

步骤 5 如果有两个 Master 节点，请在 Master2 节点上同样执行步骤 3 和步骤 4。

步骤 6 在 Master1 节点执行如下命令初始化环境变量。

source *客户端安装目录*/**bigdata_env**

步骤 7 如果当前集群已启用 Kerberos 认证，执行以下命令认证当前用户。如果当前集群未启用 Kerberos 认证，则无需执行此命令。

kinit ***MRS 集群用户***

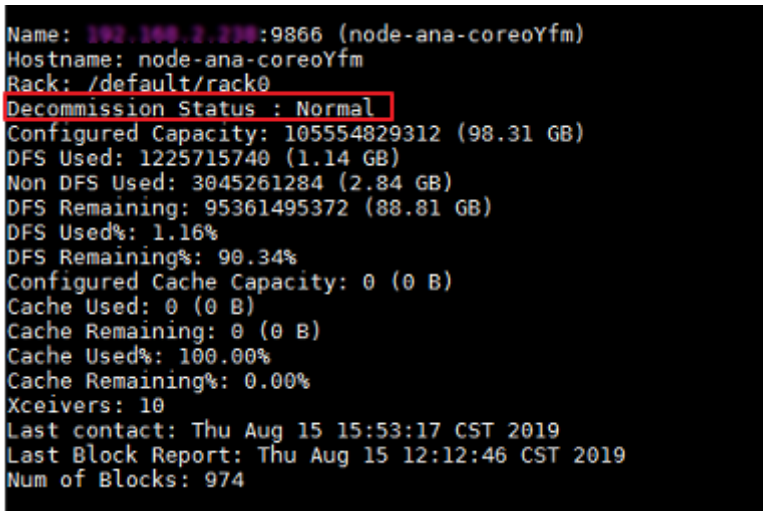
例如, **kinit admin**

步骤 8 在 Master1 节点执行如下命令刷新 HDFS 的黑名单。

hdfs dfsadmin -refreshNodes

步骤 9 使用命令 **hdfs dfsadmin -report** 来查看各个 DataNode 的状态，确认中查到的 IP 对应的 DataNode 已经恢复为 **Normal** 状态。

图9-1 DataNode 的状态



步骤 10 进入服务实例界面。

MRS Manager 界面操作：

登录 MRS Manager，在 MRS Manager 页面，选择“服务管理 > HDFS > 实例”。

FusionInsight Manager 界面操作：

对于 MRS 3.x 及后续版本集群：可登录 FusionInsight Manager。选择“集群 > 服务 > HDFS > 实例”。

登录 MRS 控制台，选择“组件管理 > HDFS > 实例”。

步骤 11 勾选一直处于退服中的 DataNode 实例，单击“更多 > 重启实例”。

步骤 12 等待重启完成，确认 DataNode 是否恢复正常。

----结束

建议与总结

尽量不要在退服（或缩容）过程中重启节点等高危操作。

参考信息

无

9.8 内存不足导致 HDFS 启动失败

问题背景与现象

重启 HDFS 后，HDFS 的状态是 Bad，且 NameNode 实例状态常常异常，并且花很久没有退出安全模式。

原因分析

1. 在 NameNode 运行日志（/var/log/Bigdata/hdfs/nn/hadoop-omm-namendoe-XXX.log）中搜索“WARN”，可以看到有大量时间在垃圾回收，如下例中耗时较长 63s。

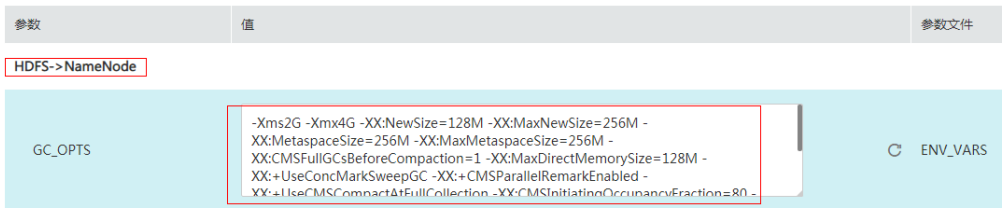
```
2017-01-22 14:52:32,641 | WARN  |
org.apache.hadoop.util.JvmPauseMonitor$Monitor@1b39fd82 | Detected pause in JVM
or host machine (eg GC): pause of approximately 63750ms
GC pool 'ParNew' had collection(s): count=1 time=0ms
GC pool 'ConcurrentMarkSweep' had collection(s): count=1 time=63924ms |
JvmPauseMonitor.java:189
```

2. 分析 NameNode 日志“/var/log/Bigdata/hdfs/nn/hadoop-omm-namendoe-XXX.log”，可以看到 NameNode 在等待块上报，且总的 Block 个数过多，如下例中是 3629 万。

```
2017-01-22 14:52:32,641 | INFO  | IPC Server handler 8 on 25000 | STATE* Safe
mode ON.
The reported blocks 29715437 needs additional 6542184 blocks to reach the
threshold 0.9990 of total blocks 36293915.
```

3. 打开 Manager 页面，查看 NameNode 的 GC_OPTS 参数配置如下：

图9-2 查看 NameNode 的 GC_OPTS 参数配置



4. NameNode 内存配置和数据量对应关系参考表 9-1。

表9-1 NameNode 内存配置和数据量对应关系

文件对象数量	参考值

文件对象数量	参考值
10,000,000	“-Xms6G -Xmx6G -XX:NewSize=512M -XX:MaxNewSize=512M”
20,000,000	“-Xms12G -Xmx12G -XX:NewSize=1G -XX:MaxNewSize=1G”
50,000,000	“-Xms32G -Xmx32G -XX:NewSize=2G -XX:MaxNewSize=3G”
100,000,000	“-Xms64G -Xmx64G -XX:NewSize=4G -XX:MaxNewSize=6G”
200,000,000	“-Xms96G -Xmx96G -XX:NewSize=8G -XX:MaxNewSize=9G”
300,000,000	“-Xms164G -Xmx164G -XX:NewSize=12G -XX:MaxNewSize=12G”

解决办法

步骤 1 按照规格修改 NameNode 的内存参数，如这里 3600 万 block，将内存参数调整为“-Xms32G -Xmx32G -XX:NewSize=2G -XX:MaxNewSize=3G”。

步骤 2 重启一个 NameNode，确认该 NameNode 可以正常启动。

步骤 3 重启另一个 NameNode，确认页面状态恢复。

----结束

9.9 ntpdate 修改时间导致 HDFS 出现大量丢块

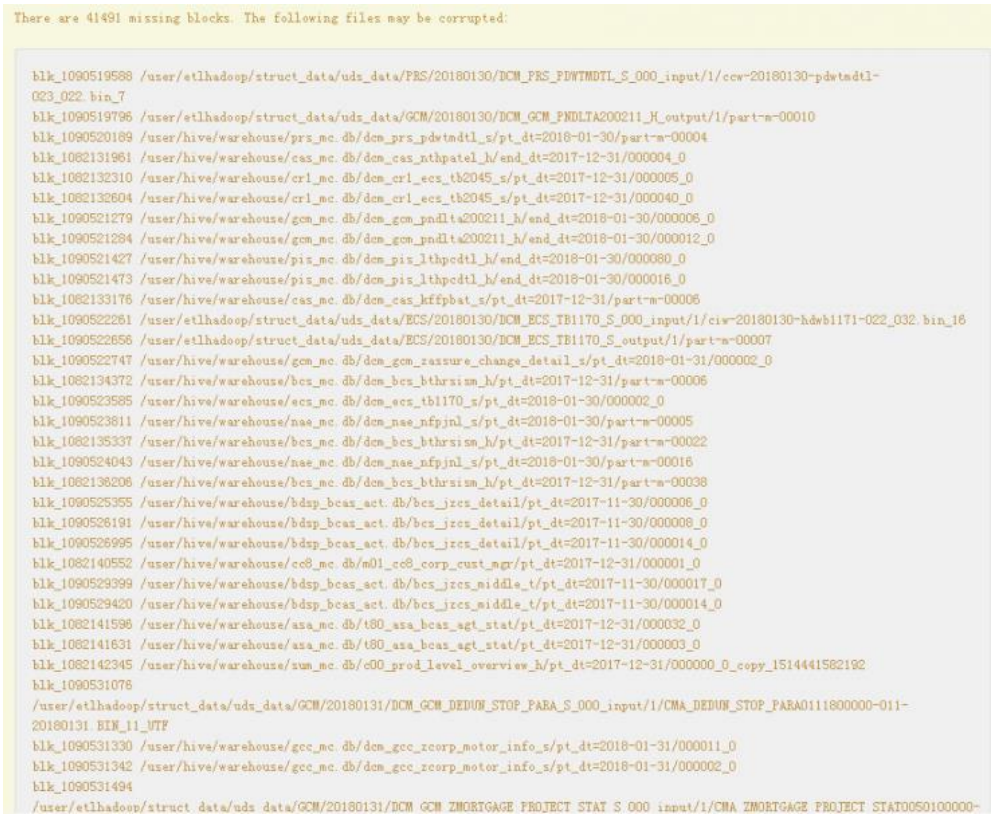
问题背景与现象

1. 用 ntpdate 修改了集群时间，修改时未停止集群，修改后 HDFS 进入安全模式，无法启动。
2. 退出安全模式后启动，hfck 检查丢了大概 1T 数据。

原因分析

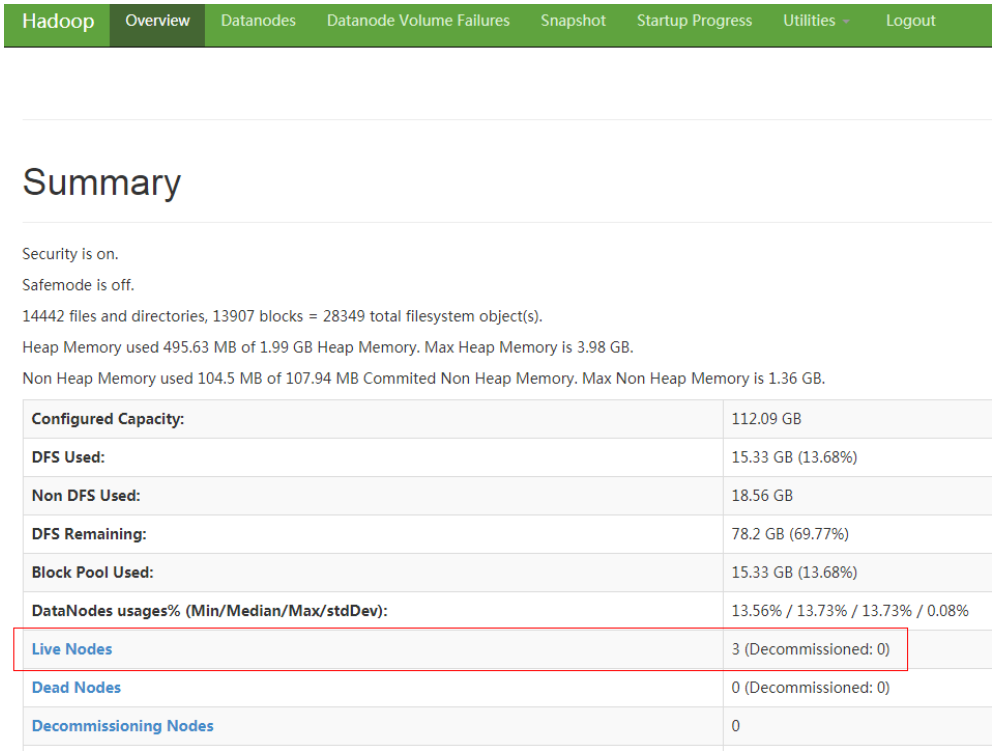
1. 查看 NameNode 原生页面发现有大量的块丢失。

图9-3 块丢失



2. 查看原生页面 Datanode Information 发现显示的 DataNode 节点数和实际的相差 10 个节点。

图9-4 查看 DataNode 节点数



3. 查看 DateNode 运行日志“/var/log/Bigdata/hdfs/dn/hadoop-omm-datanode-主机名.log”，发现如下错误信息。

   重要错误信息 Clock skew too great

图9-5 DateNode 运行日志错误

```
    at org.apache.hadoop.ipc.Client.call(Client.java:1486)
    at org.apache.hadoop.ipc.Client.call(Client.java:1447)
    at org.apache.hadoop.ipc.ProtobufRpcEngine$Invoker.invoke(ProtobufRpcEngine.java:229)
    at com.sun.proxy.$Proxy14.versionRequest(Unknown Source)
    at org.apache.hadoop.hdfs.protocolPB.DatanodeProtocolClientSideTranslatorPB.versionRequest(DatanodeProtocolClientSideTranslatorPB.java:273)
    at org.apache.hadoop.hdfs.server.datanode.BPServiceActor.retrieveNamespaceInfo(BPServiceActor.java:187)
    at org.apache.hadoop.hdfs.server.datanode.BPServiceActor.connectToNNAndHandshake(BPServiceActor.java:237)
    at org.apache.hadoop.hdfs.server.datanode.BPServiceActor.run(BPServiceActor.java:689)
    at java.lang.Thread.run(Thread.java:745)
Caused by: GSSException: No valid credentials provided (Mechanism level: Clock skew too great (37))
    at sun.security.jgss.krb5.Krb5Context.initSecContext(Krb5Context.java:770)
    at sun.security.jgss.GSSContextImpl.initSecContext(GSSContextImpl.java:248)
    at sun.security.jgss.GSSContextImpl.initSecContext(GSSContextImpl.java:179)
    at com.sun.security.sasl.gsskerb.GssKrb5Client.evaluateChallenge(GssKrb5Client.java:192)
    ... 20 more
Caused by: KrbException: Clock skew too great (37)
    at sun.security.krb5.KrbKdcRep.check(KrbKdcRep.java:88)
    at sun.security.krb5.KrbTgsRep.<init>(KrbTgsRep.java:87)
    at sun.security.krb5.KrbTgsReq.getReply(KrbTgsReq.java:259)
    at sun.security.krb5.KrbTgsReq.sendAndGetCreds(KrbTgsReq.java:270)
    at sun.security.krb5.internal.CredentialsUtil.serviceCreds(CredentialsUtil.java:302)
    at sun.security.krb5.internal.CredentialsUtil.acquireServiceCreds(CredentialsUtil.java:120)
    at sun.security.krb5.Credentials.acquireServiceCreds(Credentials.java:458)
    at sun.security.jgss.krb5.Krb5Context.initSecContext(Krb5Context.java:693)
```

解决办法

步骤 1　修改在原生页面查看不到的 10 个数据节点的时间。

步骤 2　在 Manager 页面重启对应的 DataNode 实例。

----结束

9.10 DataNode 概率性出现 CPU 占用接近 100%，导致节点丢失（ssh 连得很慢或者连不上）

问题背景与现象

DataNode 概率性出现 CPU 占用接近 100%，导致节点丢失。

图9-6 DataNode 出现 CPU 占用接近 100%

原因分析

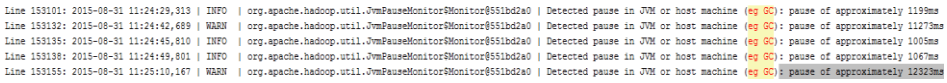
1. DataNode 有许多写失败的日志。

图9-7 DataNode 写失败的日志

```
2015-08-31 11:29:34,184 | ERROR | DataXceiver for client DFSClient_NONMAPREDUCE_1675952887_23 at /192.168.8.40:44514 [Receiving block
BP-125271511-192.168.8.29-1440656260530:blk_1074766997_1034914] | TSP21:25009:DataXceiver error processing WRITE_BLOCK operation  src:
/192.168.8.40:44514 dst: /192.168.8.64:25009 | DataXceiver.java:258
java.io.IOException: Premature EOF from inputStream
        at org.apache.hadoop.io.IOUtils.readFully(IOUtils.java:194)
        at org.apache.hadoop.hdfs.protocol.datatransfer.PacketReceiver.doReadFully(PacketReceiver.java:213)
        at org.apache.hadoop.hdfs.protocol.datatransfer.PacketReceiver.doRead(PacketReceiver.java:134)
        at org.apache.hadoop.hdfs.protocol.datatransfer.PacketReceiver.receiveNextPacket(PacketReceiver.java:109)
        at org.apache.hadoop.hdfs.server.datanode.BlockReceiver.receivePacket(BlockReceiver.java:446)
        at org.apache.hadoop.hdfs.server.datanode.BlockReceiver.receiveBlock(BlockReceiver.java:707)
        at org.apache.hadoop.hdfs.server.datanode.DataXceiver.writeBlock(DataXceiver.java:748)
        at org.apache.hadoop.hdfs.protocol.datatransfer.Receiver.opWriteBlock(Receiver.java:124)
        at org.apache.hadoop.hdfs.protocol.datatransfer.Receiver.processOp(Receiver.java:71)
        at org.apache.hadoop.hdfs.server.datanode.DataXceiver.run(DataXceiver.java:240)
        at java.lang.Thread.run(Thread.java:745)
2015-08-31 11:29:35,147 | INFO  | DataXceiver for client DFSClient_NONMAPREDUCE_-402997805_1 at /192.168.8.30:59449 [Sending block BP-
125271511-192.168.8.29-1440656260530:blk_1074181856_446655] | src: /192.168.8.64:25009, dest: /192.168.8.30:59449, bytes: 16826, op:
HDFS_READ, cliID: DFSClient_NONMAPREDUCE_-402997805_1, offset: 0, srvID: 9d1d30a5-046d-438b-83c9-2c6c54c6bd12, blockid: BP-125271511-
192.168.8.29-1440656260530:blk_1074181856_446655, duration: 78832 | BlockSender.java:738
2015-08-31 11:29:35,269 | INFO  | org.apache.hadoop.util.JvmPauseMonitor$Monitor@551bd2a0 | Detected pause in JVM or host machine (eg
GC): pause of approximately 7480ms
No GCs detected | JvmPauseMonitor.java:172
2015-08-31 11:29:36,985 | INFO  | org.apache.hadoop.util.JvmPauseMonitor$Monitor@551bd2a0 | Detected pause in JVM or host machine (eg
GC): pause of approximately 1215ms
No GCs detected | JvmPauseMonitor.java:172
2015-08-31 11:29:43,067 | INFO  | DataXceiver for client DFSClient_NONMAPREDUCE_1675952887_23 at /192.168.8.33:35530 [Receiving block
BP-125271511-192.168.8.29-1440656260530:blk_1074767006_1034923] | Exception for BP-125271511-192.168.8.29-
1440656260530:blk_1074767006_1034923 | BlockReceiver.java:742
java.io.IOException: Premature EOF from inputStream
```

2. 短时间内写入大量文件导致这种情况，因此 DataNode 内存不足。

图9-8 写入大量文件导致 DataNode 内存不足



解决办法

步骤 1 检查 DataNode 内存配置，以及机器剩余内存是否充足。

步骤 2 增加 DataNode 内存，并重启 DataNode。

----结束

9.11 单 NameNode 长期故障，如何使用客户端手动 checkpoint

问题背景与现象

在备 NameNode 长期异常的情况下，会积攒大量的 editlog，此时如果重启 HDFS 或者主 NameNode，主 NameNode 会读取大量的未合并的 editlog，导致耗时启动较长，甚至启动失败。

原因分析

备 NameNode 会周期性做合并 editlog，生成 fsimage 文件的过程叫做 checkpoint。备 NameNode 在新生成 fsimage 后，会将 fsimage 传递到主 NameNode。

说明

由于“备 NameNode 会周期性做合并 editlog”，因此当备 NameNode 异常时，无法合并 editlog，因此主 NameNode 在下次启动的时候，需要加载较多 editlog，需要大量内存，并且耗时较长。

合并元数据的周期由以下参数确定，即如果 NameNode 运行 30 分钟或者 HDFS 操作 100 万次，均会执行 checkpoint。

- dfs.namenode.checkpoint.period：checkpoint 周期，默认 1800s。
- dfs.namenode.checkpoint.txns：执行指定操作次数后执行 checkpoint，默认 1000000。

解决办法

在重启前，主动执行异常 checkpoint 合并主 NameNode 的元数据。

步骤 1 停止业务。

步骤 2 获取主 NameNode 的主机名。

步骤 3 在客户端执行如下命令：

source /opt/client/bigdata_env

kinit *组件用户*

说明：“/opt/client”需要换为实际客户端的安装路径。

步骤 4 执行如下命令，让主 NameNode 进入安全模式，其中 linux22 换为主 NameNode 的主机名。

hdfs dfsadmin -fs linux22:25000 -safemode enter

```
linux16:/opt/fi_client #  hdfs dfsadmin -fs linux22:25000   -safemode enter
17/04/26 18:38:30 WARN fs.FileSystem: "linux22:25000" is a deprecated filesystem name. Use "hdfs://linux22:25000/" instead.
17/04/26 18:38:32 INFO hdfs.PeerCache: SocketCache disabled.
Safe mode is ON
```

步骤 5 执行如下命令，在主 NameNode，合并 editlog。

hdfs dfsadmin -fs linux22:25000 -saveNamespace

```
linux16:/opt/fi_client #  hdfs dfsadmin -fs linux22:25000   -saveNamespace
17/04/26 18:38:54 WARN fs.FileSystem: "linux22:25000" is a deprecated filesystem name. Use "hdfs://linux22:25000/" instead.
17/04/26 18:38:56 INFO hdfs.PeerCache: SocketCache disabled.
Save namespace successful
```

步骤 6 执行如下命令，让主 NameNode 离开安全模式。

hdfs dfsadmin -fs linux22:25000 -safemode leave

```
linux16:/opt/fi_client #  hdfs dfsadmin -fs linux22:25000  -safemode leave
17/04/26 18:39:07 WARN fs.FileSystem: "linux22:25000" is a deprecated filesystem name. Use "hdfs://linux22:25000/" instead.
17/04/26 18:39:09 INFO hdfs.PeerCache: SocketCache disabled.
Safe mode is OFF
```

步骤 7 检查是否真的合并完成。

cd /srv/BigData/namenode/current

检查先产生的 fsimage 是否是当前时间的，若是则表示已经合并完成

```
-rw------- 1 omm wheel 1048576 Apr 26 18:43 edits_inprogress_0000000000002083018
-rw------- 1 omm wheel  736657 Apr 26 15:46 fsimage_0000000000002071390
-rw------- 1 omm wheel      62 Apr 26 15:46 fsimage_0000000000002071390.md5
-rw------- 1 omm wheel  736657 Apr 26 16:46 fsimage_0000000000002075405
-rw------- 1 omm wheel      62 Apr 26 16:46 fsimage_0000000000002075405.md5
-rw------- 1 omm wheel  736410 Apr 26 17:46 fsimage_0000000000002079398
-rw------- 1 omm wheel      62 Apr 26 17:46 fsimage_0000000000002079398.md5
-rw------- 1 omm wheel       8 Apr 26 18:42 seen_txid
linux-20:/srv/BigData/namenode/current #
linux-20:/srv/BigData/namenode/current #
```

----结束

9.12 文件读写常见故障

问题背景与现象

当用户在 HDFS 上执行写操作时，出现“Failed to place enough replicas:expected…”信息。

原因分析

- DataNode 的数据接受器不可用。

  此时 DataNode 会有如下日志：

```
2016-03-17 18:51:44,721 | WARN |
org.apache.hadoop.hdfs.server.datanode.DataXceiverServer@5386659f |
```

```
hadoopc1h2:25009:DataXceiverServer: | DataXceiverServer.java:158
java.io.IOException: Xceiver count 4097 exceeds the limit of concurrent
xcievers: 4096
at
org.apache.hadoop.hdfs.server.datanode.DataXceiverServer.run(DataXceiverServer.
java:140)
at java.lang.Thread.run(Thread.java:745)
```

- DataNode 的磁盘空间不足。
- DataNode 的心跳有延迟。

解决办法

- 如果 DataNode 的数据接收器不可用，通过在 Manager 页面，增加 HDFS 参数“dfs.datanode.max.transfer.threads”的值解决。
- 如果没有足够的硬盘空间或者 CPU，试着增加新的数据节点或确保资源是可用的（磁盘空间或 CPU）。
- 如果网络问题，确保网络是可用的。

9.13 文件最大打开句柄数设置太小导致读写文件异常

问题背景与现象

文件最大打开句柄数设置太小，导致文件句柄不足。写文件到 HDFS 很慢，或者写文件失败。

原因分析

1. DataNode 日志“/var/log/Bigdata/hdfs/dn/hadoop-omm-datanode-XXX.log”，存在异常提示 java.io.IOException: Too many open files。

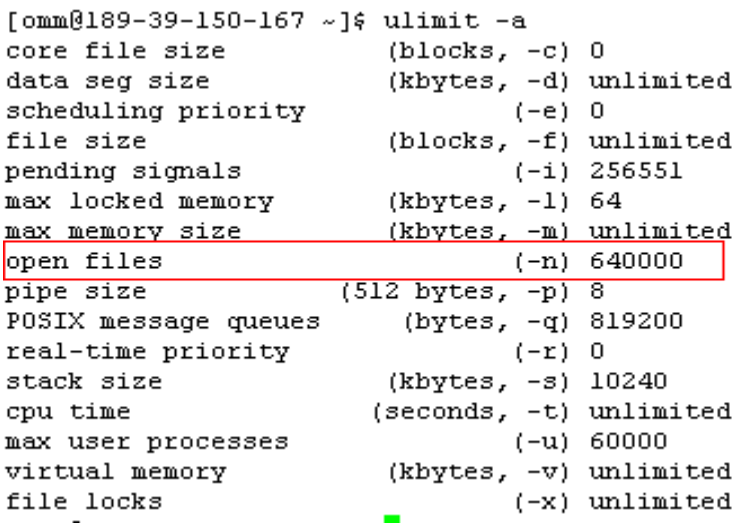
```
2016-05-19 17:18:59,126 | WARN  |
org.apache.hadoop.hdfs.server.datanode.DataXceiverServer@142ff9fa |
YSDN12:25009:DataXceiverServer:  |
org.apache.hadoop.hdfs.server.datanode.DataXceiverServer.run(DataXceiverServer.
java:160)
java.io.IOException: Too many open files
at sun.nio.ch.ServerSocketChannelImpl.accept0(Native Method)
at sun.nio.ch.ServerSocketChannelImpl.accept(ServerSocketChannelImpl.java:241)
at sun.nio.ch.ServerSocketAdaptor.accept(ServerSocketAdaptor.java:100)
at org.apache.hadoop.hdfs.net.TcpPeerServer.accept(TcpPeerServer.java:134)
at
org.apache.hadoop.hdfs.server.datanode.DataXceiverServer.run(DataXceiverServer.
java:137)
at java.lang.Thread.run(Thread.java:745)
```

2. 如果某个 DataNode 日志中打印“Too many open files”，说明该节点文件句柄不足，导致打开文件句柄失败，然后就会重试往其他 DataNode 节点写数据，最终表现为写文件很慢或者写文件失败。

解决办法

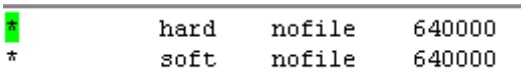
步骤 1 执行 **ulimit -a** 命令查看有问题节点文件句柄数最多设置是多少，如果很小，建议修改成 640000。

图9-9 查看文件句柄数



步骤 2 执行 **vi /etc/security/limits.d/90-nofile.conf** 命令编辑这个文件，修改文件句柄数设置。如果没有这个文件，可以新建一个文件，并按照下图内容修改。

图9-10 修改文件句柄数



步骤 3 重新打开一个终端窗口，用 **ulimit -a** 命令查看是否修改成功，如果没有，请重新按照上述步骤重新修改。

步骤 4 从 Manager 页面重启 DataNode 实例。

----结束

9.14 客户端写文件 close 失败

问题背景与现象

客户端写文件 close 失败，客户端提示数据块没有足够副本数。

客户端日志：

```
2015-05-27 19:00:52.811 [pool-2-thread-3] ERROR:
/tsp/nedata/collect/UGW/ugwufdr/20150527/10/6_20150527105000_20150527105500_SR5S14_
1432723806338_128_11.pkg.tmp1432723806338 close hdfs sequence file fail
(SequenceFileInfoChannel.java:444)
java.io.IOException: Unable to close file because the last block does not have
enough number of replicas.
at org.apache.hadoop.hdfs.DFSOutputStream.completeFile(DFSOutputStream.java:2160)
at org.apache.hadoop.hdfs.DFSOutputStream.close(DFSOutputStream.java:2128)
at
org.apache.hadoop.fs.FSDataOutputStream$PositionCache.close(FSDataOutputStream.java
:70)
at org.apache.hadoop.fs.FSDataOutputStream.close(FSDataOutputStream.java:103)
at
com.xxx.pai.collect2.stream.SequenceFileInfoChannel.close(SequenceFileInfoChannel.j
ava:433)
at
com.xxx.pai.collect2.stream.SequenceFileWriterToolChannel$FileCloseTask.call(Sequen
ceFileWriterToolChannel.java:804)
at
com.xxx.pai.collect2.stream.SequenceFileWriterToolChannel$FileCloseTask.call(Sequen
ceFileWriterToolChannel.java:792)
at java.util.concurrent.FutureTask.run(FutureTask.java:262)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1145)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:615)
at java.lang.Thread.run(Thread.java:745)
```

原因分析

1. HDFS 客户端开始写 Block。

   例如：HDFS 客户端是在 2015-05-27 18:50:24,232 开始写 /20150527/10/6_20150527105000_20150527105500_SR5S14_1432723806338_128_11.pkg.tmp1432723806338 的。其中分配的块是 blk_1099105501_25370893。

   ```
   2015-05-27 18:50:24,232 | INFO  | IPC Server handler 30 on 25000 | BLOCK*
   allocateBlock:
   /20150527/10/6_20150527105000_20150527105500_SR5S14_1432723806338_128_11.pkg.tm
   p1432723806338. BP-1803470917-192.168.57.33-1428597734132
   blk_1099105501_25370893{blockUCState=UNDER_CONSTRUCTION, primaryNodeIndex=-1,
   replicas=[ReplicaUnderConstruction[[DISK]DS-b2d7b7d0-f410-4958-8eba-
   6deecbca2f87:NORMAL|RBW], ReplicaUnderConstruction[[DISK]DS-76bd80e7-ad58-49c6-
   bf2c-03f91caf750f:NORMAL|RBW]]} |
   org.apache.hadoop.hdfs.server.namenode.FSNamesystem.saveAllocatedBlock(FSNamesy
   stem.java:3166)
   ```

2. 写完之后 HDFS 客户端调用了 fsync。

   ```
   2015-05-27 19:00:22,717 | INFO  | IPC Server handler 22 on 25000 | BLOCK* fsync:
   20150527/10/6_20150527105000_20150527105500_SR5S14_1432723806338_128_11.pkg.tmp
   1432723806338 for DFSClient_NONMAPREDUCE_-120525246_15 |
   org.apache.hadoop.hdfs.server.namenode.FSNamesystem.fsync(FSNamesystem.java:380
   5)
   ```

3. HDFS 客户端调用 close 关闭文件，NameNode 收到客户端的 close 请求之后就会检查最后一个块的完成状态，只有当有足够的 DataNode 上报了块完成才可用关闭文件，检查块完成的状态是通过 checkFileProgress 函数检查的，打印如下：

```
2015-05-27 19:00:27,603 | INFO  | IPC Server handler 44 on 25000 | BLOCK*
checkFileProgress: blk_1099105501_25370893{blockUCState=COMMITTED,
primaryNodeIndex=-1, replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-
4813-ae9a-34a0714ec3a3:NORMAL|RBW], ReplicaUnderConstruction[[DISK]DS-f863e30f-
ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached minimal replication 1
|
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesys
tem.java:3197)
2015-05-27 19:00:28,005 | INFO  | IPC Server handler 45 on 25000 | BLOCK*
checkFileProgress: blk_1099105501_25370893{blockUCState=COMMITTED,
primaryNodeIndex=-1, replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-
4813-ae9a-34a0714ec3a3:NORMAL|RBW], ReplicaUnderConstruction[[DISK]DS-f863e30f-
ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached minimal replication 1
|
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesys
tem.java:3197)
2015-05-27 19:00:28,806 | INFO  | IPC Server handler 63 on 25000 | BLOCK*
checkFileProgress: blk_1099105501_25370893{blockUCState=COMMITTED,
primaryNodeIndex=-1, replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-
4813-ae9a-34a0714ec3a3:NORMAL|RBW], ReplicaUnderConstruction[[DISK]DS-f863e30f-
ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached minimal replication 1
|
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesys
tem.java:3197)
2015-05-27 19:00:30,408 | INFO  | IPC Server handler 43 on 25000 | BLOCK*
checkFileProgress: blk 1099105501 25370893{blockUCState=COMMITTED,
primaryNodeIndex=-1, replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-
4813-ae9a-34a0714ec3a3:NORMAL|RBW], ReplicaUnderConstruction[[DISK]DS-f863e30f-
ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached minimal replication 1
|
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesys
tem.java:3197)
2015-05-27 19:00:33,610 | INFO  | IPC Server handler 37 on 25000 | BLOCK*
checkFileProgress: blk_1099105501_25370893{blockUCState=COMMITTED,
primaryNodeIndex=-1, replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-
4813-ae9a-34a0714ec3a3:NORMAL|RBW], ReplicaUnderConstruction[[DISK]DS-f863e30f-
ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached minimal replication 1
|
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesys
tem.java:3197)
2015-05-27 19:00:40,011 | INFO  | IPC Server handler 37 on 25000 | BLOCK*
checkFileProgress: blk_1099105501_25370893{blockUCState=COMMITTED,
primaryNodeIndex=-1, replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-
4813-ae9a-34a0714ec3a3:NORMAL|RBW], ReplicaUnderConstruction[[DISK]DS-f863e30f-
ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached minimal replication 1
|
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesys
tem.java:3197)
```

4. NameNode 打印了多次 checkFileProgress 是由于 HDFS 客户端多次尝试 close 文件，但是由于当前状态不满足要求，导致 close 失败， HDFS 客户端 retry 的次数是由参数 dfs.client.block.write.locateFollowingBlock.retries 决定的，该参数默认是 5，所以在 NameNode 的日志中看到了 6 次 checkFileProgress 打印。
5. 但是再过 0.5s 之后，DataNode 就上报块已经成功写入。

```
2015-05-27 19:00:40,608 | INFO  | IPC Server handler 60 on 25000 | BLOCK*
addStoredBlock: blockMap updated: 192.168.10.21:25009 is added to
blk_1099105501_25370893{blockUCState=COMMITTED, primaryNodeIndex=-1,
replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-4813-ae9a-
34a0714ec3a3:NORMAL|RBW], ReplicaUnderConstruction[[DISK]DS-f863e30f-ce5b-48cc-
9cca-72f64c558adc:NORMAL|RBW]]} size 11837530 |
org.apache.hadoop.hdfs.server.blockmanagement.BlockManager.logAddStoredBlock(Bl
ockManager.java:2393)
2015-05-27 19:00:48,297 | INFO  | IPC Server handler 37 on 25000 | BLOCK*
addStoredBlock: blockMap updated: 192.168.10.10:25009 is added to
blk_1099105501_25370893 size 11837530 |
org.apache.hadoop.hdfs.server.blockmanagement.BlockManager.logAddStoredBlock(Bl
ockManager.java:2393)
```

6. DataNode 上报块写成功通知延迟的原因可能有：网络瓶颈导致、CPU 瓶颈导致。
7. 如果此时再次调用 close 或者 close 的 retry 的次数增多，那么 close 都将返回成功。建议适当增大参数 dfs.client.block.write.locateFollowingBlock.retries 的值，默认值为 5 次，尝试的时间间隔为 400ms、800ms、1600ms、3200ms、6400ms，12800ms，那么 close 函数最多需要 25.2 秒才能返回。

解决办法

步骤 1 规避办法：

可以通过调整客户端参数 dfs.client.block.write.locateFollowingBlock.retries 的值来增加 retry 的次数，可以将值设置为 6，那么中间睡眠等待的时间为 400ms、800ms、1600ms、3200ms、6400ms、12800ms，也就是说 close 函数最多要 50.8 秒才能返回。

----结束

备注说明

一般出现上述现象，说明集群负载很大，通过调整参数只是临时规避这个问题，建议还是降低集群负载。例如：避免把所有 CPU 都分配 MR 跑任务。

9.15 文件错误导致上传文件到 HDFS 失败

问题背景与现象

用 **hadoop dfs -put** 把本地文件拷贝到 HDFS 上，有报错。

上传部分文件后，报错失败，从 NameNode 原生页面看，临时文件大小不再变化。

原因分析

1. 查看 NameNode 日志“/var/log/Bigdata/hdfs/nn/hadoop-omm-namenode-主机名.log”，发现该文件一直在被尝试写，直到最终失败。

```
2015-07-13 10:05:07,847 | WARN  |
org.apache.hadoop.hdfs.server.namenode.LeaseManager$Monitor@36fea922 | DIR*
NameSystem.internalReleaseLease: Failed to release lease for file
```

```
/hive/order/OS_ORDER._8.txt._COPYING_. Committed blocks are waiting to be
minimally replicated. Try again later. | FSNamesystem.java:3936
2015-07-13 10:05:07,847 | ERROR |
org.apache.hadoop.hdfs.server.namenode.LeaseManager$Monitor@36fea922 | Cannot
release the path /hive/order/OS_ORDER._8.txt._COPYING_ in the lease [Lease.
Holder: DFSClient_NONMAPREDUCE_-1872896146_1, pendingcreates: 1] |
LeaseManager.java:459
org.apache.hadoop.hdfs.protocol.AlreadyBeingCreatedException: DIR*
NameSystem.internalReleaseLease: Failed to release lease for file
/hive/order/OS_ORDER._8.txt._COPYING_. Committed blocks are waiting to be
minimally replicated. Try again later.
at FSNamesystem.internalReleaseLease(FSNamesystem.java:3937)
```

2. 根因分析：被上传的文件损坏，因此会上传失败。
3. 验证办法：cp 或者 scp 被拷贝的文件，也会失败，确认文件本身已损坏。

解决办法

步骤 1 文件本身损坏造成的此问题，采用正常文件进行上传。

----结束

9.16 界面配置 dfs.blocksize 后 put 数据，block 大小还是原来的大小

问题背景与现象

界面配置“dfs.blocksize”，将其设置为 268435456，put 数据，block 大小还是原来的大小。

原因分析

客户端的“hdfs-site.xml”文件中的 dfs.blocksize 大小没有更改，以客户端配置为准。

解决办法

步骤 1 确保“dfs.blocksize”为 512 的倍数。

步骤 2 重新下载安装客户端或者更改客户端配置。

步骤 3 dfs.blocksize 是客户端配置，以客户端为准。若客户端不配置，以服务端为准。

----结束

9.17 读取文件失败，FileNotFoundException

问题背景与现象

有 MapReduce 任务所有 map 任务均成功，但 reduce 任务失败，查看日志发现报异常“FileNotFoundException...No lease on...File does not exist”。

```
Error: org.apache.hadoop.ipc.RemoteException(java.io.FileNotFoundException): No
lease on
/user/sparkhive/warehouse/daas/dsp/output/_temporary/1/_temporary/attempt_147979905
3892_17075_r_000007_0/part-r-00007 (inode 6501287): File does not exist. Holder
DFSClient_attempt_1479799053892_17075_r_000007_0_-1463597952_1 does not have any
open files.
at
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkLease(FSNamesystem.java:33
50)
at
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.completeFileInternal(FSNamesyst
em.java:3442)
at
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.completeFile(FSNamesystem.java:
3409)
at
org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.complete(NameNodeRpcServer
.java:789)
```

原因分析

FileNotFoundException...No lease on...File does not exist，该日志说明文件在操作的过程中被删除了。

1. 搜索 HDFS 的 NameNode 的审计日志（Active NameNode 的 /var/log/Bigdata/audit/hdfs/nn/hdfs-audit-namenode.log）搜索文件名，确认文件的创建时间。
2. 搜索文件创建到出现异常时间范围的 NameNode 的审计日志，搜索该文件是否被删除或者移动到其他目录。
3. 如果该文件没有被删除或者移动，可能是该文件的父目录，或者更上层目录被删除或者移动，需要继续搜索上层目录。如本样例中，是文件的父目录的父目录被删除。

   ```
   2017-05-31 02:04:08,286 | INFO  | IPC Server handler 30 on 25000 | allowed=true
   ugi=appUser@HADOOP.COM (auth:TOKEN) ip=/192.168.1.22    cmd=delete
   src=/user/sparkhive/warehouse/daas/dsp/output/_temporary     dst=null
   perm=null proto=rpc | FSNamesystem.java:8189
   ```

说明

- 如上日志说明：192.168.1.22 节点的 appUser 用户删除了 /user/sparkhive/warehouse/daas/dsp/output/_temporary。
- 可以使用 **zgrep "文件名" *.zip** 命令搜索 zip 包的内容。

解决办法

步骤 1 需要排查业务，确认为何该文件或者文件的父目录被删除。

----结束

9.18 HDFS 写文件失败，item limit of / is exceeded

问题背景与现象

客户端或者上层组件日志报往 HDFS 的某目录写文件失败，报错为

The directory item limit of /tmp is exceeded: limit=5 items=5。

原因分析

1. 查看客户端或者 NameNode 运行日志“/var/log/Bigdata/hdfs/nn/hadoop-omm-namenode-XXX.log”存在异常提示 The directory item limit of /tmp is exceeded:。该错误的含义为/tmp 目录的文件数超过 1048576 的限制。

```
2018-03-14 11:18:21,625 | WARN  | IPC Server handler 62 on 25000 | DIR*
NameSystem.startFile: /tmp/test.txt The directory item limit of /tmp is
exceeded: limit=1048576 items=1048577 | FSNamesystem.java:2334
```

2. 该限制是 dfs.namenode.fs-limits.max-directory-items 参数，定义单个目录下不含递归的最大目录数或者文件数，默认值 1048576，取值范围 1～6400000。

解决办法

步骤 1 确认该目录不含递归拥有 100 万以上文件目录是否正常，如果正常，可以将 HDFS 的参数 dfs.namenode.fs-limits.max-directory-items 调大并且重启 HDFS NameNode 生效。

步骤 2 如果该目录下拥有 100 万文件不正常，需要清理不需要的文件。

----结束

9.19 调整 shell 客户端日志级别

- **临时调整**，关闭该 shell 客户端窗口后，日志会还原为默认值。
  a. 执行 **export HADOOP_ROOT_LOGGER** 命令可以调整客户端日志级别。
  b. 执行 **export HADOOP_ROOT_LOGGER=*日志级别*,console**，可以调整 shell 客户端的日志级别。

     export HADOOP_ROOT_LOGGER=DEBUG,console，调整为 DEBUG。

     export HADOOP_ROOT_LOGGER=ERROR,console，调整为 ERROR。
- **永久调整**

a. 在 HDFS 客户端环境变量配置文件“/opt/client/HDFS/component_env”（其中“/opt/client”需要改为实际客户端路径）增加“export HADOOP_ROOT_LOGGER=*日志级别*,console”。

b. 执行 **source /opt/client/bigdata_env**。

c. 重新执行客户端命令。

9.20 读文件失败 No common protection layer

问题背景与现象

shell 客户端或者其他客户端操作 HDFS 失败，报“**No common protection layer between client and server**”。

在集群外的机器，执行任意 hadoop 命令，如 **hadoop fs -ls** /均失败，最底层的报错为"**No common protection layer between client and server**"。

```
2017-05-13 19:14:19,060 | ERROR | [pool-1-thread-1] |  Server startup failure  |
org.apache.sqoop.core.SqoopServer.initializeServer(SqoopServer.java:69)
org.apache.sqoop.common.SqoopException: MAPRED_EXEC_0028:Failed to operate HDFS -
Failed to get the file /user/loader/etl_dirty_data_dir status
       at org.apache.sqoop.job.mr.HDFSClient.fileExist(HDFSClient.java:85)
...
       at java.lang.Thread.run(Thread.java:745)
Caused by: java.io.IOException: Failed on local exception: java.io.IOException:
Couldn't setup connection for loader/hadoop@HADOOP.COM to
loader37/10.162.0.37:25000; Host Details : local host is: "loader37/10.162.0.37";
destination host is: "loader37":25000;
       at org.apache.hadoop.net.NetUtils.wrapException(NetUtils.java:776)
...
       ... 10 more
Caused by: java.io.IOException: Couldn't setup connection for
loader/hadoop@HADOOP.COM to loader37/10.162.0.37:25000
       at org.apache.hadoop.ipc.Client$Connection$1.run(Client.java:674
       ... 28 more
Caused by: javax.security.sasl.SaslException: No common protection layer between
client and server
       at
com.sun.security.sasl.gsskerb.GssKrb5Client.doFinalHandshake(GssKrb5Client.java:251)
...
       at org.apache.hadoop.ipc.Client$Connection.setupIOstreams(Client.java:720)
```

原因分析

1. HDFS 的客户端和服务端数据传输走的 rpc 协议，该协议有多种加密方式，由 hadoop.rpc.protection 参数控制。
2. 如果客户端和服务端的 hadoop.rpc.protection 参数的配置值不一样，即会报 **No common protection layer between client and server** 错误。

说明

hadoop.rpc.protection 参数表示数据可通过以下任一方式在节点间进行传输。

- privacy：指数据在鉴权及加密后再传输。这种方式会降低性能。
- authentication：指数据在鉴权后直接传输，不加密。这种方式能保证性能但存在安全风险。
- integrity：指数据直接传输，即不加密也不鉴权。 为保证数据安全，请谨慎使用这种方式。

解决办法

步骤 1 重新下载客户端，如果是应用程序，更新应用程序中的配置文件。

----结束

9.21 HDFS 目录配额（quota）不足导致写文件失败

问题背景与现象

给某目录设置 quota 后，往目录中写文件失败，出现如下问题“**The DiskSpace quota of** /tmp/tquota2 is **exceeded**”

```
[omm@189-39-150-115 client]$ hdfs dfs -put switchuser.py  /tmp/tquota2
put: The DiskSpace quota of /tmp/tquota2 is exceeded: quota = 157286400 B = 150 MB
but diskspace consumed = 402653184 B = 384 MB
```

可能原因

目录配置的剩余的空间小于写文件实际需要的空间。

原因分析

1. HDFS 支持设置某目录的配额，即限制某目录下的文件最多占用空间大小，例如如下命令是设置/tmp/tquota 目录最多写入 150MB 的文件（文件大小*副本数）。

   hadoop dfsadmin -setSpaceQuota 150M /tmp/tquota2

2. 使用如下命令可以查看目录设置的配额情况，SPACE_QUOTA 是设置的空间配额，REM_SPACE_QUOTA 是当前剩余的空间配额。

   hdfs dfs -count -q -h -v /tmp/tquota2

   图9-11 查看目录设置的配额

```
hdfs dfs -count -q -h -v /tmp/tquota2
QUOTA REM_QUOTA SPACE_QUOTA REM_SPACE_QUOTA  DIR_COUNT   FILE_COUNT CONTENT_SIZE PATHNAME
none  inf         150M        150M              1            0            0  /tmp/tquota2
```

3. 日志分析，如下日志说明写入文件需要消耗 384M，但是当前的空间配额是 150M，因此空间不足。写文件前，需要的剩余空间是：块大小*副本数，128M*3 副本=384M。

```
[omm@189-39-150-115 client]$
[omm@189-39-150-115 client]$ hdfs dfs -put switchuser.py  /tmp/tquota2
put: The DiskSpace quota of /tmp/tquota2 is exceeded: quota = 157286400 B = 150
MB but diskspace consumed = 402653184 B = 384 MB
```

解决办法

步骤 1 增加配额大小，即重新设置目录的配额大小。

hadoop dfsadmin -setSpaceQuota 150G /*目录名*

步骤 2 清空配额。

hdfs dfsadmin -clrSpaceQuota /*目录名*

----结束

9.22 执行 balance 失败，Source and target differ in block-size

问题背景与现象

执行 distcp 跨集群拷贝文件时，出现部分文件拷贝失败“ Source and target differ in block-size. Use -pb to preserve block-sizes during copy. ”

```
Caused by: java.io.IOException: Check-sum mismatch between
hdfs://10.180.144.7:25000/kylin/kylin_default_instance_prod/parquet/f2e72874-f01c-
45ff-b219-207f3a5b3fcb/c769cd2d-575a-4459-837b-
a19dd7b20c27/339114721280/0.parquettar and
hdfs://10.180.180.194:25000/kylin/kylin_default_instance_prod/parquet/f2e72874-
f01c-45ff-b219-207f3a5b3fcb/.distcp.tmp.attempt_1523424430246_0004_m_000019_2.
Source and target differ in block-size. Use -pb to preserve block-sizes during copy.
Alternatively, skip checksum-checks altogether, using -skipCrc. (NOTE: By skipping
checksums, one runs the risk of masking data-corruption during file-transfer.)
at
org.apache.hadoop.tools.mapred.RetriableFileCopyCommand.compareCheckSums(RetriableF
ileCopyCommand.java:214)
```

可能原因

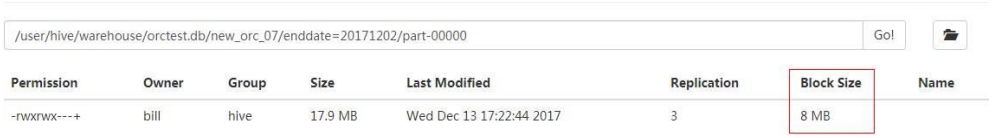
distcp 默认拷贝文件时不记录原 block 大小导致在原文件 block.size 不是 128M 时校验失败，需要在 distcp 命令增加-pb 参数。

原因分析

1. HDFS 在写的时候有设置块大小，默认 128M，某些组件或者业务程序写入的文件可能不是 128M，如 8M。

```
<name>dfs.blocksize</name>
<value>134217728</value>
```

图9-12 某些组件或者业务程序写入的文件大小



2. distcp 从源集群读文件后写入新集群，默认是使用的 MapReduce 任务中的 dfs.blocksize，默认 128M。
3. 在 distcp 写完文件后，会基于块的物理大小做校验，因为该文件在新旧集群中 block.size 不一致，因此拆分大小不一致，导致校验失败。

   如以上文件，在旧集群是 17.9/8MB = 3 个 block， 在新集群 17.9/128M = 1 个 block. 因此实际在磁盘的物理大小因分割而导致校验失败。

解决办法

distcp 时，增加**-pb** 参数。该参数作用为 distcp 时候保留 block 大小，确保新集群写入文件 blocksize 和老集群一致。

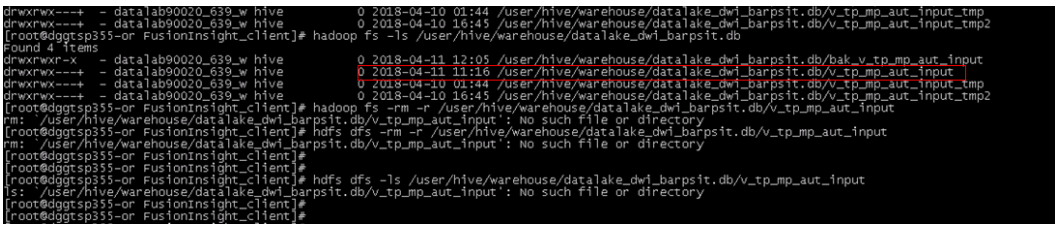
图9-13 distcp 时保留 block 大小

9.23 查询或者删除文件失败，父目录可以看见此文件（不可见字符）

问题背景与现象

使用 HDFS 的 shell 客户端查询或者删除文件失败，父目录可以看见此文件。

图9-14 父目录文件列表



原因分析

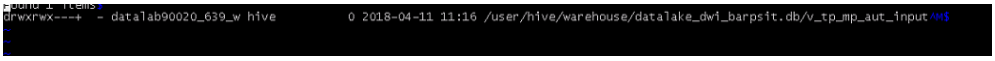
可能是该文件写入时有异常，写入了不可见字符。可以将该文件名重定向写入本地文本中，使用 **vi** 命令打开。

hdfs dfs -ls *父目录* **> /tmp/t.txt**

vi /tmp/t.txt

然后输入命令“**:set list**”将文件名的不可见字符显示出来。如这里显示出文件名中包含“**^M**”不可见字符。

图9-15 显示不可见字符

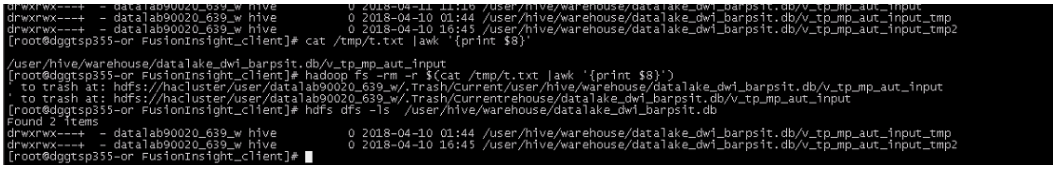


解决办法

步骤 1 使用 shell 命令读到文本中记录的该文件名，确认如下命令输出的是该文件在 HDFS 中的全路径。

cat /tmp/t.txt |awk '{print $8}'

图9-16 文件路径



步骤 2 使用如下命令删除该文件。

hdfs dfs -rm $(cat /tmp/t.txt |awk '{print $8}')

步骤 3 查看确认该文件已被删除。

hdfs dfs -ls *父目录*

----结束

9.24 非 HDFS 数据残留导致数据分布不均衡

问题背景与现象

数据出现不均衡，某磁盘过满而其他磁盘未写满。

HDFS DataNode 数据存储目录配置为“/export/data1/dfs--/export/data12/dfs”，看到的现象是大量数据都是存储到了“/export/data1/dfs”，其他盘的数据比较均衡。

原因分析

磁盘为卸载重装，有一个目录在上次卸载时未卸载干净，即添加的磁盘，未格式化，残留历史垃圾数据。

解决办法

手动清理未卸载干净的数据。

9.25 客户端安装在数据节点导致数据分布不均衡

问题背景与现象

HDFS 的 DataNode 数据分布不均匀，在某节点上磁盘使用率很高，甚至达到 100%，其他节点空闲很多。

原因分析

客户端安装在该节点，根据 HDFS 数据副本机制，第一个副本会存放在本地机器，最终导致节点磁盘被占满，而其他节点空闲很多。

解决办法

步骤 1 针对已有不平衡的数据，执行 balance 脚本均衡数据。

/opt/client/HDFS/hadoop/sbin/start-balancer.sh -threshold 10

其中“/opt/client”是实际的客户端安装目录。

步骤 2 针对新写入数据，将客户端安装在没有安装 DataNode 的节点。

----结束

9.26 节点内 DataNode 磁盘使用率不均衡处理指导

问题背景与现象

单个节点内 DataNode 的各磁盘使用率不均匀。

例如：

```
189-39-235-71:~ # df -h
Filesystem  Size  Used Avail Use% Mounted on
/dev/xvda  360G  92G  250G  28% /
/dev/xvdb  700G  900G  200G  78% /srv/BigData/hadoop/data1
/dev/xvdc  700G  900G  200G  78% /srv/BigData/hadoop/data2
/dev/xvdd  700G  900G  200G  78% /srv/BigData/hadoop/data3
/dev/xvde  700G  900G  200G  78% /srv/BigData/hadoop/data4
```

```
/dev/xvdf  10G   900G   890G  2% /srv/BigData/hadoop/data5
189-39-235-71:~ #
```

可能原因

部分磁盘故障，更换为新盘，因此新盘使用率低。

增加了磁盘个数，如原先 4 个数据盘，现扩容为 5 个数据盘。

原因分析

DataNode 节点内写 block 磁盘时，有两种策略“轮询”和“优先写剩余磁盘空间多的磁盘”，默认是“轮询”。

参数说明：dfs.datanode.fsdataset.volume.choosing.policy

可选值：

- 轮询：org.apache.hadoop.hdfs.server.datanode.fsdataset.RoundRobinVolumeChoosingPolicy
- 优先写剩余空间多的磁盘：org.apache.hadoop.hdfs.server.datanode.fsdataset.AvailableSpaceVolumeChoosingPolicy

解决办法

将 DataNode 选择磁盘策略的参数 dfs.datanode.fsdataset.volume.choosing.policy 的值改为：org.apache.hadoop.hdfs.server.datanode.fsdataset.AvailableSpaceVolumeChoosingPolicy，保存并重启受影响的服务或实例。

让 DataNode 根据磁盘剩余空间大小，优先选择磁盘剩余空间多的节点存储数据副本。

说明

- 针对新写入到本 DataNode 的数据会优先写磁盘剩余空间多的磁盘。
- 部分磁盘使用率较高，依赖业务逐渐删除在 HDFS 中的数据（老化数据）来逐渐降低。

9.27 执行 balance 常见问题定位方法

问题 1：报没权限（Access denied）执行 balance

问题详细：执行 start-balancer.sh，“hadoop-root-balancer-主机名.out”日志显示“Access denied for user test1. Superuser privilege is required”

```
cat /opt/client/HDFS/hadoop/logs/hadoop-root-balancer-host2.out
Time Stamp               Iteration#  Bytes Already Moved  Bytes Left To Move  Bytes
Being Moved
INFO: Watching file:/opt/client/HDFS/hadoop/etc/hadoop/log4j.properties for changes
with interval : 60000
org.apache.hadoop.ipc.RemoteException(org.apache.hadoop.security.AccessControlExcep
tion): Access denied for user test1.
```

```
Superuser privilege is required
at
org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkSuperuserPrivilege(
FSPermissionChecker.java:122)
at
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkSuperuserPrivilege(FSNames
ystem.java:5916)
```

问题根因：

执行 balance 需要使用管理员账户

解决方法

- 安全版本
  使用 hdfs 或者其他属于 supergroup 组的用户认证后，执行 balance
- 普通版本
  执行 HDFS 的 balance 命令前，需要在客户端执行 su - hdfs 命令。

问题 2：执行 balance 失败，/system/balancer.id 文件异常

问题详细：

在 HDFS 客户端启动一个 Balance 进程，该进程被异常停止后，再次执行 Balance 操作，操作会失败。

```
org.apache.hadoop.ipc.RemoteException(org.apache.hadoop.protocol.RecoveryInProgress
Exception): Failed to APPEND_FILE /system/balancer.id for DFSClient because lease
recovery is in progress. Try again later.
```

问题根因：

通常，HDFS 执行 Balance 操作结束后，会自动释放“/system/balancer.id”文件，可再次正常执行 Balance。

但在上述场景中，由于第一次的 Balance 操作是被异常停止的，所以第二次进行 Balance 操作时，“/system/balancer.id”文件仍然存在，则会触发 append /system/balancer.id 操作，进而导致 Balance 操作失败。

解决方法

方法 1：等待硬租期超过 1 小时后，原有客户端释放租约，再执行第二次 Balance 操作。

方法 2：删除 HDFS 中的“/system/balancer.id”文件，再执行下次 Balance 操作。

9.28 HDFS 显示磁盘空间不足，其实还有 10%磁盘空间

问题背景与现象

1. 出现“HDFS 磁盘空间使用率超过阈值”告警。
2. 查看 HDFS 页面，查看磁盘空间使用率非常高。

原因分析

HDFS 中配置了 dfs.datanode.du.reserved.percentage 参数：每个磁盘的保留空间所占磁盘百分比。DataNode 会保留这么多可用空间，以备其他组件如 Yarn 的 NodeManager 运行计算时，或者预留升级时使用。

因为预留了 10%的磁盘，当磁盘使用率达到 90%的时候，HDFS 的 DataNode 即会认为没有可用磁盘空间。

解决办法

步骤 1 扩容，在 HDFS DataNode 磁盘到 80%，即需要及时扩容。

步骤 2 如不能及时扩容，需要删除 HDFS 中的不需要数据，释放磁盘空间。

----结束

9.29 普通集群在 Core 节点安装 hdfs 客户端，使用时报错

用户问题

普通集群在 Core 节点新建用户安装使用客户端报错。

问题现象

普通集群在 Core 节点新建用户安装使用客户端报错如下：

```
2020-03-14 19:16:17,166 WARN shortcircuit.DomainSocketFactory: error creating
DomainSocket
java.net.ConnectException: connect(2) error: Permission denied when trying to
connect to '/var/run/MRS-HDFS/dn_socket'
at org.apache.hadoop.net.unix.DomainSocket.connect0(Native Method)
at org.apache.hadoop.net.unix.DomainSocket.connect(DomainSocket.java:256)
at
org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory.createSocket(DomainSocketFa
ctory.java:168)
at
org.apache.hadoop.hdfs.client.impl.BlockReaderFactory.nextDomainPeer(BlockReaderFac
tory.java:799)
...
```

原因分析

用户使用 **useradd** 命令来创建用户，此用户默认用户组不包含“ficommmon”用户组，导致在使用 hdfs 的 get 命令的时候出现上述报错。

处理步骤

使用命令 **usermod -a -G ficommon username** 为用户添加用户组“ficommon”。

9.30 集群外节点安装客户端使用 hdfs 上传文件失败

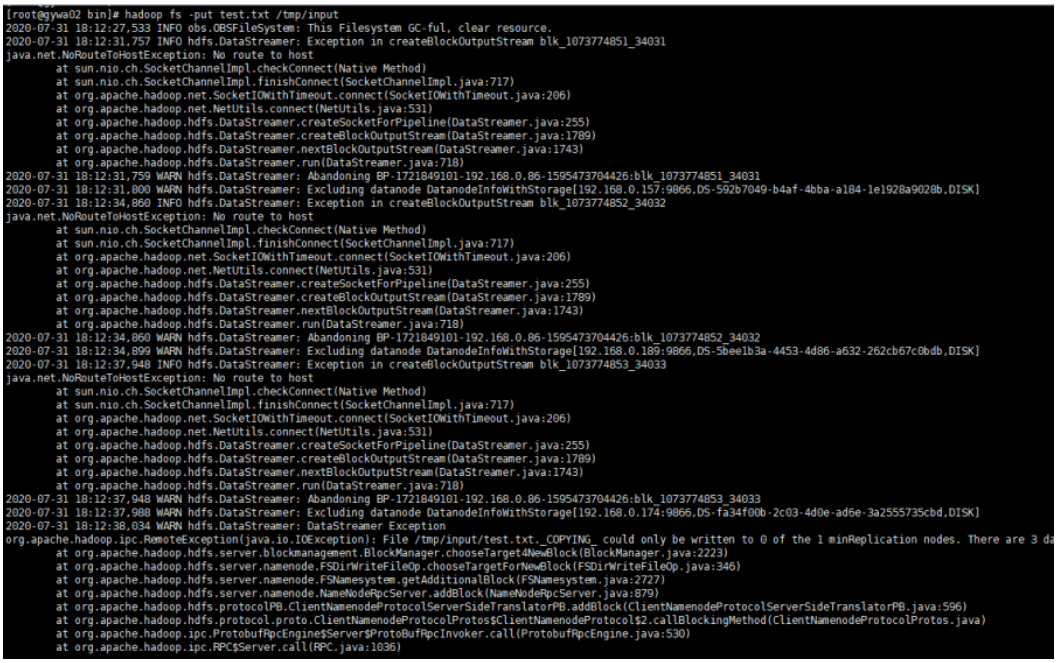
用户问题

集群外节点安装客户端使用 hdfs 上传文件失败

问题现象

在集群节点上安装客户端，在该客户端使用 hdfs 命令上传一个文件，报如下错误：

图9-17 上传文件报错



原因分析

从错误截图可以看到报错是 no route to host，且报错信息里面有 192.168 的 ip，也即客户端节点到集群的 DN 节点的内网路由不通，导致上传文件失败。

处理步骤

在客户端节点的客户端目录下，找到 HDFS 的客户端配置目录 hdfs-site.xml 文件，在配置文件中增加配置项 dfs.client.use.datanode.hostname，并将该配置设置为 true。

9.31 HDFS 写并发较大时，报副本不足的问题

问题背景与现象

用户运行作业时写文件到 HDFS，偶现写文件失败的情况。

操作日志如下：

```
105 | INFO  | IPC Server handler 23 on 25000 | IPC Server handler 23 on 25000, call
org.apache.hadoop.hdfs.protocol.ClientProtocol.addBlock from 192.168.1.96:47728
Call#1461167 Retry#0 | Server.java:2278
java.io.IOException: File /hive/warehouse/000000_0.835bf64f-4103 could only be
replicated to 0 nodes instead of minReplication (=1).  There are 3 datanode(s)
running and 3 node(s) are excluded in this operation.
```

原因分析

- HDFS 写文件的预约机制：无论文件是 10M 还是 1G，开始写的每个块都会被预约 128M。如果需要写入一个 10M 的文件，HDFS 会预约一个块来写，当文件写完后，这个块只占实际大小 10M，释放多余预约的 118M 空间。如果需要写入一个 1G 的文件，HDFS 还是会预约一个块来写，这个块写完后再开启下一个块，文件写完后，实际占用 1G 磁盘，释放多余预约的空间。
- 该异常通常是因为业务写文件的并发量太高，预约写 Block 的磁盘空间不足，导致写文件失败。

解决办法

步骤 1 登录 HDFS 的 WebUI 页面，进入 DataNode 的 JMX 页面。

1. 在 HDFS 原生界面，选择 Datanodes 页面。
2. 找到对应的 DataNode 节点，单击 Http Address 地址进入 DataNode 详情。
3. 将 url 的“datanode.html”改为“jmx”就能获取到 DataNode 的 JMX 信息。

步骤 2 搜索“XceiverCount”指标，当该指标的值*Block 块的大小超过 DataNode 磁盘的容量，就说明预约写 Block 的磁盘空间不足。

步骤 3 发生该问题，通常有以下两种方法来解决：

方法一：降低业务的并发度。

方法二：减少业务写文件的数目，将多个文件合并成一个文件来写。

----结束

9.32 HDFS 客户端无法删除超长目录

问题背景与现象

执行 **hadoop fs -rm -r -f obs://**<*obs_path*>命令，删除 OBS 超长目录出现如下报错：

```
2022-02-28 17:12:45,605 INFO internal.RestStorageService: OkHttp cost 19 ms to
apply http request
2022-02-28 17:12:45,606 WARN internal.RestStorageService: Request failed, Response
code: 400; Request ID: 0000017F3F9A8545401491602FC8CAD9; Request path:
http://wordcount01-
fcq.obs.xxx.ulanqab.xxx.com/user%2Froot%2F.Trash%2FCurrent%2Ftest1%2F12345678901234
5678901234567890123456789012345678901234567890123456789012345678901234567890123456
8901234567890123456789012345678901234567890123456789012345678901234567890123456789
1234567890123456789012345678901234567890123456789012345678901234567890123456789012
3456789012345678901234567890123456789012345678901234567890123456789012345678901234
5678901234567890123456789012345678901234567890123456789012345678901234567890123456
7890123456789012345678901234567890123456789012345678901234567890123456789012345678
9012345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890123456789012345678901234567890123456789012
3456789012345678901234567890123456789012345678901234567890123456789012345678901234
5678901234567890123456789012345678901234567890123456789012345678901234567890123456
7890123456789012345678901234567890123456789012345678901234567890123456789012345678
9012345678901234567890123456789012345678901234567890123456789012345678901234567890
123456
2022-02-28 17:12:45,606 WARN services.AbstractClient:
Storage|1|HTTP+XML|getObjectMetadata||||2022-02-28 17:12:45|2022-02-28
17:12:45|||400|
2022-02-28 17:12:45,607 INFO log.AccessLogger: 2022-02-28 17:12:45
605|com.obs.services.internal.RestStorageService|executeRequest|560|OkHttp cost 19
ms to apply http request
2022-02-28 17:12:45
606|com.obs.services.internal.RestStorageService|handleThrowable|221|Request  failed,
Response code: 400; Request ID: 0000017F3F9A8545401491602FC8CAD9; Request path:
http://wordcount01-
fcq.obs.xxx.ulanqab.xxx.com/user%2Froot%2F.Trash%2FCurrent%2Ftest1%2F12345678901234
5678901234567890123456789012345678901234567890123456789012345678901234567890123456
8901234567890123456789012345678901234567890123456789012345678901234567890123456789
1234567890123456789012345678901234567890123456789012345678901234567890123456789012
3456789012345678901234567890123456789012345678901234567890123456789012345678901234
5678901234567890123456789012345678901234567890123456789012345678901234567890123456
7890123456789012345678901234567890123456789012345678901234567890123456789012345678
9012345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890123456789012345678901234567890123456789012
3456789012345678901234567890123456789012345678901234567890123456789012345678901234
5678901234567890123456789012345678901234567890123456789012345678901234567890123456
7890123456789012345678901234567890123456789012345678901234567890123456789012345678
9012345678901234567890123456789012345678901234567890123456789012345678901234567890
123456
2022-02-28 17:12:45
606|com.obs.services.AbstractClient|doActionWithResult|404|Storage|1|HTTP+XML|getOb
jectMetadata||||2022-02-28 17:12:45|2022-02-28 17:12:45|||400|
```

原因分析

使用 rm 命令从 HDFS 中删除某些内容时，文件或目录不会立即被清除，它们将被移动到回收站 Current 目录（/user/*${username}*/.Trash/current）中。

解决办法

使用 skipTrash 命令可以跳过 HDFS 回收站，直接删除。使用前先设置 HDFS 客户端配置项“dfs.client.skipTrash.enabled=true”。

步骤 1 以 **root** 用户登录集群任一 Master 节点。

步骤 2 执行如下命令编辑 HDFS 用到的 hdfs-site.xml 文件。

vim *客户单安装目录*/**HDFS/hadoop/etc/hadoop/hdfs-site.xml**

步骤 3 在 hdfs-site.xml 文件中增加如下内容。

```
<property>
<name>dfs.client.skipTrash.enabled</name>
<value>true</value>
</property>
```

步骤 4 执行以下命令直接删除 OBS 超长目录。

hadoop fs -rm -r -f -skipTrash obs://<*obs_path*>

步骤 5 登录集群其他 Master 节点，执行步骤 2~步骤 4，直到集群所有 Master 节点操作完成。

----结束

9.33 NameNode 节点存在 ALM-12027 主机 PID 使用率超过阈值告警

问题背景与现象

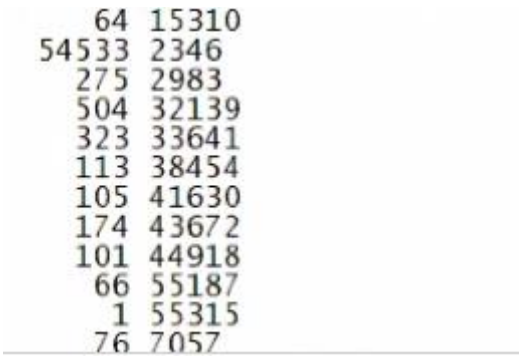
3.1.2 及之前的 3.x 版本集群，NameNode 节点存在 ALM-12027 主机 PID 使用率超过阈值告警，节点 Java 进程可能出现“unable to create new native thread”报错。

原因分析

1. 使用以下命令统计节点进程的线程数 并排序。

   ps -efT | awk '{print $2}' |sort -n |uniq -c |sort -n

   执行后结果如下：



2. 查看启动线程数最多的进程，案例中进程 2346 为 NameNode 进程，启动了 5.4 万线程，且持续增长。
3. 多次打印对应进程的 jstack 日志，根据 jstack 日志信息发现，NameNode 存在大量线程处于 WAITING，且长期不释放。

```
Line 200:    at java.util.concurrent.locks.LockSupport.park(LockSupport.java:175)
Line 215:    at java.util.concurrent.locks.LockSupport.park(LockSupport.java:175)
Line 230:    at java.util.concurrent.locks.LockSupport.park(LockSupport.java:175)
Line 245:    at java.util.concurrent.locks.LockSupport.park(LockSupport.java:175)
Line 260:    at java.util.concurrent.locks.LockSupport.park(LockSupport.java:175)
Line 275:    at java.util.concurrent.locks.LockSupport.park(LockSupport.java:175)
```

```
"Thread-121224" #653415 daemon prio=5 os_prio=0 tid=0x00007fb25b753000 nid=0x50ea waiting on condition [0x00007fb16a287000]
   java.lang.Thread.State: WAITING (parking)
    at sun.misc.Unsafe.park(Native Method)
    - parking to wait for  <0x000000073f5e8838> (a java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject)
    at java.util.concurrent.locks.LockSupport.park(LockSupport.java:175)
    at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.await(AbstractQueuedSynchronizer.java:2045)
    at java.util.concurrent.LinkedBlockingQueue.take(LinkedBlockingQueue.java:442)
    at java.util.concurrent.ThreadPoolExecutor.getTask(ThreadPoolExecutor.java:1074)
    at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1134)
    at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
    at java.lang.Thread.run(Thread.java:748)

   Locked ownable synchronizers:
    - None
```

结合以上问题分析如下：NameNode 存在内置机制，根据 WARN 日志信息自动开启 DEBUG 日志，在环境中由于选择副本失败，导致一直启动 Debug 日志，不停的修改 log4j，修改组件的 log4j 后进程会自动加载该配置文件，此时就会有新的线程自动产生，长时间后就会触发该告警。

出现这种情况时，将内置机制关闭，禁止自动修改日志级别即可恢复。

解决办法

步骤 1 分别登录到集群主备 NameNode 节点，执行以下命令备份脚本。

cd $BIGDATA_HOME/FusionInsight_Current/*_*_NameNode/install/hadoop/sbin

cp hdfs-namenode-period-check.sh /tmp

步骤 2 在主备 NameNode 节点编辑 hdfs-namenode-period-check.sh 文件。

vi hdfs-namenode-period-check.sh

在 main 方法中将“checkBlockplacementLog”注释掉，例如：

```
main()
{
    Log $INFO "start period check"
    checkHaState
    checkDefaultFS
    checkAutoBalancer
    checkFsMonitorDirectory
    checkAutoMover
    checkAutoDatamove
    checkAutoNodeLabelrefresh
    checkJournalNodeSync
    checkCheckpoint
    checkCleanAcls
    checkSssdMonitor
    checkOperationCollecter
    checkMapReduceDistributedCache
    #checkBlockplacementLog
```

```
    checkAutoDiskBalancer
}
```

步骤 3 保存文件后，登录 Manager，选择“集群 > 服务 > HDFS > 实例”，勾选所有 NameNode 实例，选择“更多 > 重启实例”。

----结束

9.34 集群出现 ALM-14012 Journalnode 数据不同步告警

问题背景与现象

MRS 集群出现 ALM-14012 Journalnode 数据不同步告警。

原因分析

1. 登录告警节点，查找日志路径“/var/log/Bigdata/hdfs/nn”下 Journalnode 实例的 startDetail.log 日志信息，发现 Journalnode 实例停止过。
2. 分别查看告警节点和其他 JournalNode 节点的“/srv/BigData/journalnode/hacluster/current”路径下最新的 edits 日志文件，发现告警节点与其他节点存在不同步的情况。

解决办法

步骤 1 登录 FusionInsight Manager，选择“集群 > 服务 > HDFS > 实例”，勾选告警发生节点对应的 Journalnode 实例，选择“更多 > 停止实例”。

步骤 2 登录告警节点，将“/srv/BigData/journalnode/hacluster/current”目录下的所有文件移动到其他新建目录下（例如“/opt/test”），保持该目录下清空状态。

步骤 3 登录 FusionInsight Manager，选择“集群 > 服务 > HDFS > 实例”，勾选停止的 Journalnode 实例，单击“启动实例”。

步骤 4 等待一段时间后，观察告警是否恢复。

----结束

9.35 由于 HDFS 块丢失导致 DataNode 退服失败

问题背景与现象

在退服 DataNode 过程中，一直提示退服失败。

原因分析

1. 查看退服失败报错日志，日志中显示总计 1564 个块，有一个块一直没法被备份。

```
[2021-02-25 09:52:46]NameNode#192.168.1.51, current underReplicatedBlocks: 1, total blocks: 1564.
[2021-02-25 09:52:50]NameNode#192.168.1.51, current underReplicatedBlocks: 1, total blocks: 1564.
[2021-02-25 09:52:55]NameNode#192.168.1.51, current underReplicatedBlocks: 1, total blocks: 1564.
[2021-02-25 09:52:59]NameNode#192.168.1.51, current underReplicatedBlocks: 1, total blocks: 1564.
[2021-02-25 09:53:03]NameNode#192.168.1.51, current underReplicatedBlocks: 1, total blocks: 1564.
[2021-02-25 09:53:07]NameNode#192.168.1.51, current underReplicatedBlocks: 1, total blocks: 1564.
[2021-02-25 09:53:11]NameNode#192.168.1.51, current underReplicatedBlocks: 1, total blocks: 1564.
[2021-02-25 09:53:16]NameNode#192.168.1.51, current underReplicatedBlocks: 1, total blocks: 1564.
[2021-02-25 09:53:20]NameNode#192.168.1.51, current underReplicatedBlocks: 1, total blocks: 1564.
[2021-02-25 09:53:20]Node 192.168.1.44 decommission failed, no data was decommissioned in 30 minutes.
[2021-02-25 09:53:20]192.168.1.44#DataNode cannot be decommissioned.
```

2. 登录集群 Master 节点，进入 HDFS 客户端，执行 **hdfs fsck /**命令查看损坏的块，并记录文件路径。

   例如：/tmp/hive-scratch/omm/_tez_session_dir/*xxx*-resources/*xxx*.jar。

   并且 HDFS 状态为“CORRUPT”

```
Erasure Coded Block Groups:
 Total size:    0 B
 Total files:   0
 Total block groups (validated):         0
 Minimally erasure-coded block groups:   0
 Over-erasure-coded block groups:        0
 Under-erasure-coded block groups:       0
 Unsatisfactory placement block groups: 0
 Average block group size:       0.0
 Missing block groups:           0
 Corrupt block groups:           0
 Missing internal blocks:        0
FSCK ended at Thu Feb 25 17:36:56 CST 2021 in 2584 milliseconds


The filesystem under path '/' is CORRUPT
```

解决办法

步骤 1 请确认该损坏的块是否可以删除。

- 是，执行步骤 2。
- 否，请联系技术支持。

步骤 2 执行以下命令进入 HDFS 客户端。

cd *HDFS 客户端安装目录*

source bigdata_env

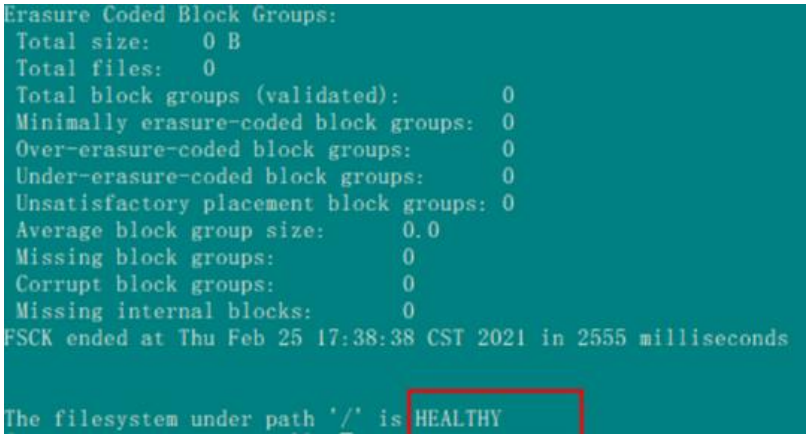
kinit *业务用户*

步骤 3 执行以下命令删除之前记录的损坏的块。

hdfs dfs -rm -skipTrash */tmp/hive-scratch/omm/_tez_session_dir/xxx-resources/xxx.jar*

步骤 4 执行命令查看 HDFS 状态是否恢复为“HEALTHY”。

hdfs fsck /



步骤 5 再次执行 DataNode 退服操作。

----结束

9.36 使用 distcp 命令拷贝空文件夹报错

问题背景与现象

通过 MRS 客户端使用以下 distcp 命令，无法从 HDFS 复制空文件夹到 OBS。

hadoop distcp -Dfs.obs.endpoint=xxx

-Dfs.obs.access.key=*xxx* **-Dfs.obs.secret.key=***xxx* **-update hdfs://hacluster/blee obs://xxx/aaa**

原因分析

如果源端（例如“blee”）为空目录，且目的端（例如“aaa”）目录不存在，系统会自动创建出来“aaa”目录，但是不会在“aaa”目录下面再创建“blee”目录。

如果源端“blee”不是空目录，且目的端“aaa”目录不存在，系统会自动创建出来“aaa”目录，并在“aaa”目录下面再创建“blee”目录，进行文件迁移。

解决办法

- 进行迁移操作时，源端目录（例如“blee”）不建议为空。
- 如果源端目录为空，在执行迁移前，需要手动创建目的端目录，即手动创建“aaa”目录。

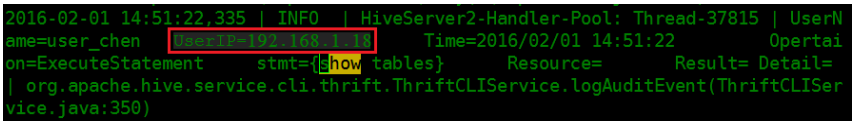
10 使用 Hive

10.1 Hive 各个日志里都存放了什么信息？

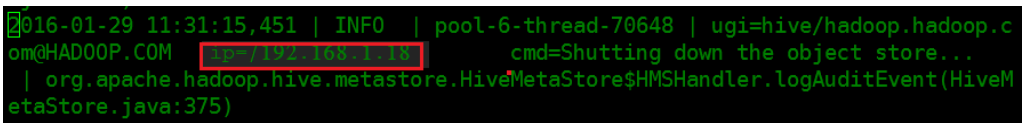
审计日志

首先，对于审计日志来说，记录了某个时间点某个用户从哪个 IP 发起对 HiveServer 或者 MetaStore 的请求以及记录执行的语句是什么。

如下的 HiveServer 审计日志，表示在 2016-02-01 14:51:22 用户 user_chen 向 HiveServer 发起了 show tables 请求，客户端 IP 为 192.168.1.18。



如下面 MetaStore 审计日志，表示在 2016-01-29 11:31:15 用户 hive 向 MetaStore 发起 shutdown 请求，客户端 ip 为 192.168.1.18。



通常情况下，审计日志对定位实际错误信息并无太大帮助。但在遇到诸如下述类问题时，需要查看审计日志：

1. 如客户端发起请求，但是迟迟未得到响应。由于不确定任务是卡在客户端还是服务端，可以通过审计日志查看。如果审计日志根本没有相关信息，那么说明卡死在客户端；如审计日志有相关打印，那么就需要去运行日志里查看程序卡在哪一步了。
2. 查看指定时间段的任务请求个数。可通过审计日志查看在指定时间段有多少个请求。

HiveServer 运行日志

简言之，HiveServer 负责接收客户端请求（SQL 语句），然后编译、执行（提交到 YARN 或运行 local MR）、与 MetaStore 交互获取元数据信息等。HiveServer 运行日志记录了一个 SQL 完整的执行过程。

通常情况下，当遇到 SQL 语句运行失败，首先需要查看 HiveServer 运行日志。

MetaStore 运行日志

通常情况下，当遇到查看 HiveServer 运行日志时，如遇到 MetaException 或者连接 MetaStore 失败，则需要查看 MetaStore 运行日志。

GC 日志查看

HiveServer 和 MetaStore 均有 GC 日志，当遇到 GC 问题可以查看 GC 日志以快速定位是否是 GC 导致。如，当遇到 HiveServer 或 MetaStore 频繁重启就需要去看下对应的 GC 日志了。

10.2 Hive 启动失败问题的原因有哪些？

Hive 启动失败最常见的原因是 metastore 实例无法连接上 DBservice。可以查看 metastore 日志中具体的错误信息。目前总结连不上 DBservice 原因主要有：

可能原因 1

DBservice 没有初始化好 Hive 的元数据库 hivemeta。

处理步骤 1

步骤 1 执行以下命令：

source /opt/Bigdata/MRS_*XXX*/install/dbservice/.dbservice_profile

gsql -h *DBservice 浮动 IP* **-p 20051 -d hivemeta -U hive -W HiveUser@**

步骤 2 如果不能正确进入交互界面，说明数据库初始化失败。如果报如下错误说明在 DBservice 所在的节点的配置文件可能丢失了 hivemeta 的配置。

```
org.postgresql.util.PSQLException: FATAL: no pg_hba.conf entry for host
"192.168.0.146", database "HIVEMETA"。
```

步骤 3 编辑“/srv/BigData/dbdata_service/data/pg_hba.conf”，在文件最后面追加 **host hivemeta hive 0.0.0.0/0 sha256** 配置。

步骤 4 执行 **source /opt/Bigdata/MRS_*XXX*/install/dbservice/.dbservice_profile** 命令配置环境变量。

步骤 5 执行 **gs_ctl -D $GAUSSDATA reload #**命令使修改后的配置生效。

----结束

可能原因 2

DBservice 的浮动 IP 配置有误，导致 metastore 节点 IP 无法正确连接浮动 IP，或者是在与该 ip 建立互信的时候失败导致 metastore 启动失败。

处理步骤 2

DBservice 的浮动 IP 配置需要同网段内没有被使用过的 ip，也就是在配置前 ping 不通的 ip，请修改 DBService 浮动 IP 配置。

10.3 安全集群执行 set 命令的时候报 Cannot modify xxx at runtime.

问题现象

执行 set 命令时报以下错误：

```
0: jdbc:hive2://192.168.1.18:21066/> set mapred.job.queue.name=QueueA;
 Error: Error while processing statement: Cannot modify mapred.job.queue.name at
list of params that are allowed to be modified at runtime (state=42000,code=1)
```

处理步骤

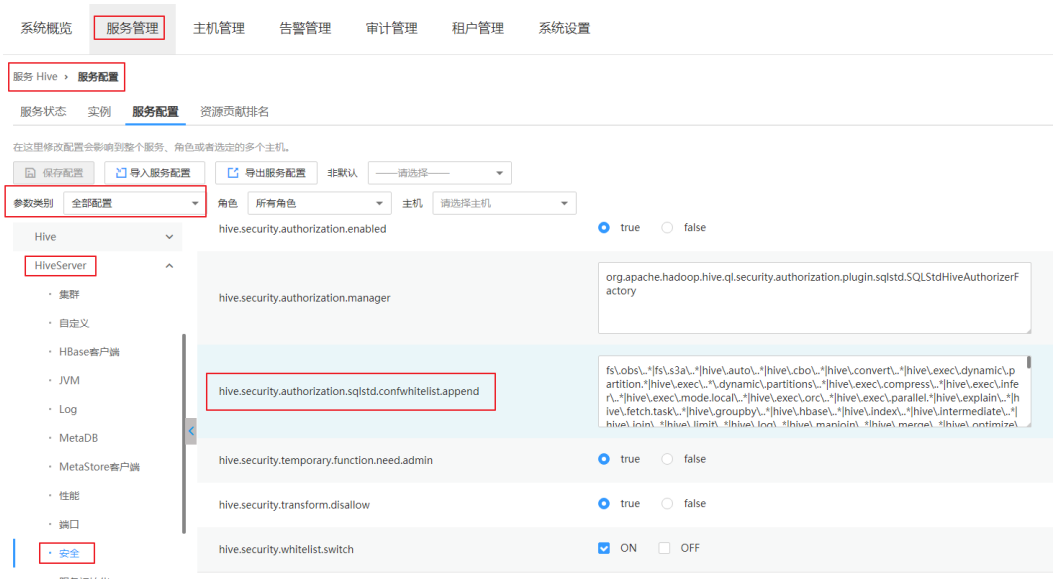
方案 1：

步骤 1 登录 Manager 界面，修改 Hive 参数。

- MRS Manager 界面操作：登录 MRS Manager 页面，选择“服务管理 > Hive > 服务配置 > 全部配置 > HiveServer > 安全”。
- FusionInsight Manager 界面操作：登录 FusionInsight Manager 页面，选择“集群 > *待操作集群的名称* > 服务 > Hive > 配置 > 全部配置 > HiveServer > 安全”。

步骤 2 将需要执行的命令参数添加到配置项 hive.security.authorization.sqlstd.confwhitelist.append 中。

步骤 3 单击保存并重启 HiveServer 后即可。如下图所示：



----结束

方案 2：

步骤 1 登录 Manager 界面，修改 Hive 参数。

- MRS Manager 界面操作：登录 MRS Manager 页面，选择“服务管理 > Hive > 服务配置 > 全部配置 > HiveServer > 安全”。
- FusionInsight Manager 界面操作：登录 FusionInsight Manager 页面，选择“集群 > *待操作集群的名称* > 服务 > Hive > 配置 > 全部配置 > HiveServer > 安全”。

步骤 2 找到选项 hive.security.whitelist.switch，选择 OFF，单击保存并重启 HiveServer 即可。

----结束

10.4 怎样在 Hive 提交任务的时候指定队列？

问题现象

怎样在 Hive 提交任务的时候指定队列？

处理步骤

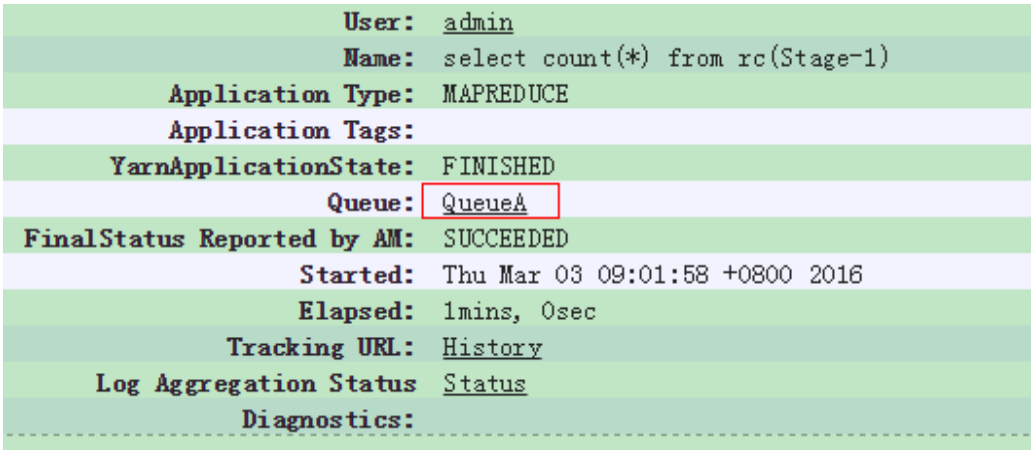
步骤 1 在执行语句前通过如下参数设置任务队列，例如，提交任务至队列 QueueA。

```
set mapred.job.queue.name=QueueA;
select count(*) from rc;
```

说明

队列的名称区分大小写，如写成 queueA，Queuea 均无效；且该队列为叶子队列，不支持提交任务到非叶子队列。

步骤 2 提交任务后，可在 Yarn 页面看到，如下任务已经提交到队列 QueueA。



----结束

10.5 客户端怎么设置 Map/Reduce 内存？

问题现象

客户端怎么设置 Map/Reduce 内存？

处理步骤

Hive 在执行 SQL 语句前，可以通过 set 命令来设置 Map/Reduce 相关客户端参数。

以下为与 Map/Reduce 内存相关的参数：

```
set mapreduce.map.memory.mb=4096;// 每个 Map Task 需要的内存量
set mapreduce.map.java.opts=-Xmx3276M; // 每个 Map Task 的 JVM 最大使用内存
set mapreduce.reduce.memory.mb=4096; // 每个 Reduce Task 需要的内存量
set mapreduce.reduce.java.opts=-Xmx3276M; // 每个 Reduce Task 的 JVM 最大使用内存
set mapred.child.java.opts=-Xms1024M -Xmx3584M;//此参数为全局参数，既对 Map 和 Reduce 统一设置
```

说明

参数设置只对当前 session 有效。

10.6 如何在导入表时指定输出的文件压缩格式

问题现象

如何在导入表时指定输出的文件压缩格式？

处理步骤

当前 Hive 支持以下几种压缩格式：

```
org.apache.hadoop.io.compress.BZip2Codec
org.apache.hadoop.io.compress.Lz4Codec
org.apache.hadoop.io.compress.DeflateCodec
org.apache.hadoop.io.compress.SnappyCodec
org.apache.hadoop.io.compress.GzipCodec
```

- 如需要全局设置，即对所有表都进行压缩，可以在 Manager 页面对 Hive 的服务配置参数进行如下全局配置：
  - hive.exec.compress.output 设置为 true
  - mapreduce.output.fileoutputformat.compress.codec 设置为 org.apache.hadoop.io.compress.BZip2Codec

说明

hive.exec.compress.output 参数必须设置为 true，才能使下边的参数选项生效。

- 如需在 session 级设置，只需要在执行命令前增加如下设置即可：

```
set hive.exec.compress.output=true;
set
```

```
mapreduce.output.fileoutputformat.compress.codec=org.apache.hadoop.io.compress.
SnappyCodec;
```

10.7 desc 描述表过长时，无法显示完整

问题现象

desc 描述表过长时，如何让描述显示完整？

处理步骤

步骤 1　启动 Hive 的 beeline 时，设置参数 maxWidth=20000 即可，例如：

```
[root@192-168-1-18 logs]# beeline --maxWidth=20000
scan complete in 3ms
Connecting to
……
Beeline version 1.1.0 by Apache Hive
```

步骤 2　（可选）通过 **beeline -help** 命令查看关于客户端显示的设置。如下：

```
  -u <database url>           the JDBC URL to connect to
  -n <username>             the username to connect as
  -p <password>             the password to connect as
  -d <driver class>            the driver class to use
  -i <init file>               script file for initialization
  -e <query>                query that should be executed
  -f <exec file>               script file that should be executed
  --hiveconf property=value      Use value for given property
  --color=[true/false]           control whether color is used for display
  --showHeader=[true/false]      show column names in query results
  --headerInterval=ROWS;        the interval between which heades are displayed
  --fastConnect=[true/false]      skip building table/column list for tab-completion
  --autoCommit=[true/false]      enable/disable automatic transaction commit
  --verbose=[true/false]          show verbose error messages and debug info
  --showWarnings=[true/false]     display connection warnings
  --showNestedErrs=[true/false]    display nested errors
  --numberFormat=[pattern]       format numbers using DecimalFormat pattern
  --force=[true/false]            continue running script even after errors
  --maxWidth=MAXWIDTH         the maximum width of the terminal
  --maxColumnWidth=MAXCOLWIDTH    the maximum width to use when displaying columns
  --silent=[true/false]          be more silent
  --autosave=[true/false]        automatically save preferences
  --outputformat=[table/vertical/csv2/tsv2/dsv/csv/tsv]  format mode for result
display
                             Note that csv, and tsv are deprecated - use csv2, tsv2
instead
  --truncateTable=[true/false]    truncate table column when it exceeds length
  --delimiterForDSV=DELIMITER    specify the delimiter for delimiter-separated
values output format (default: |)
  --isolation=LEVEL             set the transaction isolation level
  --nullemptystring=[true/false]  set to true to get historic behavior of printing
null as empty string
```

```
 --socketTimeOut=n            socket connection timeout interval, in second. The
default value is 300.
```

----结束

10.8 增加分区列后再 insert 数据显示为 NULL

问题现象

1. 执行如下命令创建表

```
create table test_table(
col1 string,
col2 string
)
PARTITIONED BY(p1 string)
STORED AS orc tblproperties('orc.compress'='SNAPPY');
```

2. 修改表结构，添加分区并插入数据

```
alter table test_table add partition(p1='a');
insert into test_table partition(p1='a') select col1,col2 from temp_table;
```

3. 修改表结构，添加列并插入数据

```
alter table test_table add columns(col3 string);
insert into test_table partition(p1='a') select col1,col2,col3 from temp_table;
```

4. 查询 test_table 表数据，返回结果中列 col3 的值全为 NULL

```
select * from test_table where p1='a'
```

5. 新添加表分区，并插入数据

```
alter table test_table add partition(p1='b');
insert into test_table partition(p1='b') select col1,col2,col3 from temp_table;
```

6. 查询 test_table 表数据，返回结果中列 col3 有不为 NULL 的值

```
select * from test_table where p1='b'
```

原因分析

在 alter table 时默认选项为 RESTRICT，RESTRICT 只会更改元数据，不会修改在此操作之前创建的 partition 的表结构，而只会修改在此之后创建的新的 partition，所以查询时旧的 partition 中的值全为 NULL。

处理步骤

add column 时加入 cascade 关键字即可，例如：

```
alter table test_table add columns(col3 string) cascade;
```

10.9 创建新用户，执行查询时报无权限

问题现象

创建了新用户，但是执行查询的时候报无权限的错。

```
Error: Error while compiling statement: FAILED: HiveAccessControlException
Permission denied: Principal [name=hive, type=USER] does not have following
privileges for operation QUERY [[SELECT] on Object [type=TABLE OR VIEW,
name=default.t1]] (state=42000,code=40000)
```

原因分析

创建的新用户没有 Hive 组件的操作权限。

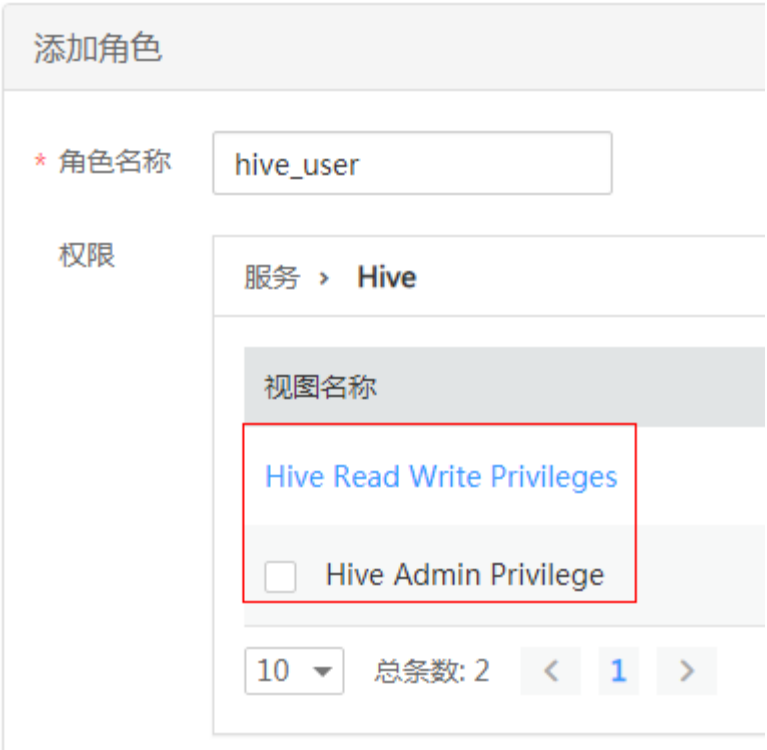
解决方案

MRS Manager 界面操作：

步骤 1 登录 MRS Manager 页面，选择“系统配置 > 角色管理 > 添加角色”。

步骤 2 输入角色名称。

步骤 3 在“权限”区域选择 Hive，出现 Hive 管理员权限和 Hive 表的读写权限。



步骤 4 选择“Hive Read Write Privileges”Hive 表的读写权限，此时显示列 Hive 中的所有数据库。

步骤 5 勾选角色需要的权限并单击“确定”完成角色创建。

步骤 6 在 MRS Manager 页面，选择“系统配置 > 用户管理”。

步骤 7 在已创建的新用户对应的“操作”列单击“修改”。

步骤 8 单击“选择添加的用户组”，如需使用 Hive 服务，必须添加 Hive 组。

步骤 9 单击“选择并绑定角色”，勾选步骤 5 中已创建的角色。

步骤 10 单击“确定”完成用户权限的配置。

----结束

FusionInsight Manager 界面操作：

步骤 1 登录 FusionInsight Manager。选择“系统 > 权限 > 角色”。

步骤 2 单击“添加角色”，输入“角色名称”和“描述”。

步骤 3 设置角色“配置资源权限”，选择“Hive 读写权限”Hive 表的读写权限，此时显示列 Hive 中的所有数据库。

步骤 4 勾选角色需要的权限并单击“确定”完成角色创建。

步骤 5 在 FusionInsight Manager 页面，选择“系统 > 权限 > 用户”。

步骤 6 在已创建的新用户对应的“操作”列单击“修改”。

步骤 7 单击“用户组”右侧的“添加”，如需使用 Hive 服务，必须添加 Hive 组。

步骤 8 单击“角色”右侧的“添加”，勾选 4 中已创建的角色。

步骤 9 单击“确定”完成用户权限的配置。

----结束

10.10 执行 SQL 提交任务到指定队列报错

问题现象

执行 SQL 提交任务到 Yarn 报如下错误：

```
Failed to submit application_1475400939788_0033 to YARN :
org.apache.hadoop.security.AccessControlException: User newtest cannot submit
applications to queue root.QueueA
```

原因分析

当前登录的用户无 YARN 队列提交权限。

解决方案

用户无 YARN 队列提交权限，需要赋予 YARN 相应队列的提交权限。在 Manager 页面的“系统 >权限 > 用户”中给用户绑定队列提交权限的角色。

10.11 执行 load data inpath 命令报错

问题现象

执行 load data inpath 报如下错误：

- 错误 1：

```
HiveAccessControlException Permission denied. Principal [name=user1, type=USER]
does not have following privileges on Object [type=DFS_URI,
name=hdfs://hacluster/tmp/input/mapdata] for operation LOAD : [OBJECT OWNERSHIP]
```

- 错误 2：

```
HiveAccessControlException Permission denied. Principal [name=user1, type=USER]
does not have following privileges on Object [type=DFS_URI,
name=hdfs://hacluster/tmp/input/mapdata] for operation LOAD : [INSERT, DELETE]
```

- 错误 3：

```
SemanticException [Error 10028]: Line 1:17 Path is not legal
''file:///tmp/input/mapdata'': Move from: file:/tmp/input/mapdata to:
hdfs://hacluster/user/hive/warehouse/tmp1 is not valid. Please check that
values for params "default.fs.name" and "hive.metastore.warehouse.dir" do not
conflict.
```

原因分析

当前登录的用户不具备操作此目录的权限或者文件目录格式不正确。

解决方案

Hive 对 load data inpath 命令有如下权限要求，请对照下述要求是否满足：

- 文件的 owner 需要为执行命令的用户。
- 当前用户需要对该文件有读、写权限。
- 当前用户需要对该文件的目录有执行权限。
- 由于 load 操作会将该文件移动到表对应的目录中，所以要求当前用户需要对表的对应目录有写权限。
- 要求文件的格式与表指定的存储格式相同。如创建表时指定 stored as rcfile，但是文件格式为 txt，则不符合要求。
- 文件必须是 HDFS 上的文件，不可以用 file://的形式指定本地文件系统上的文件。
- 文件名不能以下横线（_）或点（.）开头，以这些开头的文件会被忽略。

  如下所示，如果用户 test_hive load 数据，正确的权限如下：

```
[root@192-168-1-18 duan]# hdfs dfs -ls /tmp/input2
16/03/21 14:45:07 INFO hdfs.PeerCache: SocketCache disabled.
Found 1 items
-rw-r--r--   3 test_hive hive          6 2016-03-21 14:44 /tmp/input2/input.txt
```

10.12 执行 load data local inpath 命令报错

问题现象

执行 load data local inpath 报如下错误：

- 错误 1：

```
HiveAccessControlException Permission denied. Principal [name=user1, type=USER]
does not have following privileges on Object [type=LOCAL_URI,
name=file:/tmp/input/mapdata] for operation LOAD : [SELECT, INSERT, DELETE]
```

- 错误 2：

```
HiveAccessControlException Permission denied. Principal [name=user1, type=USER]
does not have following privileges on Object [type=LOCAL_URI,
name=file:/tmp/input/mapdata] for operation LOAD : [OBJECT OWNERSHIP]
```

- 错误 3：

```
SemanticException Line 1:23 Invalid path ''/tmp/input/mapdata'': No files
matching path file:/tmp/input/mapdata
```

原因分析

当前登录的用户不具备操作此目录的权限或者在 HiveServer 所在节点上没有此目录。

解决方案

说明

通常不建议使用本地文件加载数据到 hive 表。 建议先将本地文件放入 HDFS，然后从集群中加载数据。

Hive 对 load data local inpath 命令有如下权限要求，请对照下述要求是否满足：

- 由于所有的命令都是发送到主 HiveServer 上去执行的，所以要求此文件在 HiveServer 节点上。
- HiveServer 进程是以操作系统上的 omm 用户启动的，所以要求 omm 用户对此文件有读权限，对此文件的目录有读、执行权限。
- 文件的 owner 需要为执行命令的用户。
- 当前用户需要对该文件有读、写权限。
- 要求文件的格式与表指定的存储格式相同。如创建表时指定 stored as rcfile，但是文件格式为 txt，则不符合要求。
- 文件名不能以下横线（_）或点（.）开头，以这些开头的文件会被忽略。

10.13 执行 create external table 报错

问题现象

执行命令：create external table xx(xx int) stored as textfile location '/tmp/aaa/aaa'，报以下错误：

```
Permission denied. Principal [name=fantasy, type=USER] does not have following
privileges on Object [type=DFS_URI, name=/tmp/aaa/aaa] for operation CREATETABLE :
[SELECT, INSERT, DELETE, OBJECT OWNERSHIP] (state=42000,code=40000)
```

原因分析

当前登录的用户不具备该目录或者其父目录的读写权限。创建外部表时，会判断当前用户对指定的目录以及该目录下其它目录和文件是否有读写权限，如果该目录不存在，会去判断其父目录，依次类推。如果一直不满足就会报权限不足。而不是报指定的目录不存在。

解决方案

请确认当前用户为路径“/tmp/aaa/aaa”的 owner 有读写权限，如果该路径不存在，确认对其父路径有读写权限。

10.14 在 beeline 客户端执行 dfs -put 命令报错

问题现象

执行命令：

dfs -put /opt/kv1.txt /tmp/kv1.txt

报以下错误：

```
Permission denied. Principal [name=admin, type=USER] does not have following
privileges onObject[type=COMMAND_PARAMS,name=[-put, /opt/kv1.txt, /tmp/kv1.txt]]
for operation DFS : [ADMIN PRIVILEGE] (state=,code=1)
```

原因分析

当前登录的用户不具备操作此命令的权限。

解决方案

如果登录的当前用户具有 admin 角色，请用 set role admin 来切换成 admin 角色操作。如果不具备 admin 角色，在 Manager 页面中给用户绑定对应角色的权限。

10.15 执行 set role admin 报无权限

问题现象

执行命令：

set role admin

报下述错误：

```
0: jdbc:hive2://192.168.42.26:21066/> set role admin;
Error: Error while processing statement: FAILED: Execution Error, return code 1
from org.apache.hadoop.hive.ql.exec.DDLTask. dmp_B doesn't belong to role admin
(state=08S01,code=1)
```

原因分析

当前登录的用户不具有 Hive 的 admin 角色的权限。

解决方案

步骤 1 登录 Manager。

- MRS 3.x 之前版本，执行步骤 7。
- MRS 3.x 及之后版本，选择“集群 > 服务 > Hive”，在服务“概览”页面右上角单击“更多”，查看“启用 Ranger 鉴权”是否置灰。
  - 是，执行步骤 2。
  - 否，执行步骤 7。

步骤 2 选择“集群 > 服务 > Ranger”，单击“基本信息”区域中的“RangerAdmin”，进入 Ranger WebUI 界面。

步骤 3 单击右上角用户名后，选择“Log Out”，退出当前用户后使用 **rangeradmin** 用户登录。

步骤 4 在首页中单击“Settings”，选择“Roles”。

步骤 5 单击“Role Name”为“admin”的角色，在“Users”区域，单击“Select User”，选择指定用户名。

步骤 6 单击 Add Users 按钮，在对应用户名所在行勾选“Is Role Admin”，单击“Save”保存配置，操作结束。

步骤 7 选择“系统 > 权限 > 角色”，添加一个拥有 Hive 管理员权限的角色。

步骤 8 在 FusionInsight Manager 页面，选择“系统 > 权限 > 用户 ”。

步骤 9 在指定用户对应的“操作”列单击“修改”。

步骤 10 为用户绑定拥有 Hive 管理员权限的角色，并单击“确定”完成权限添加。

----结束

10.16 通过 beeline 创建 UDF 时候报错

问题现象

执行命令：

create function fn_test3 as 'test.MyUDF' using jar 'hdfs:///tmp/udf2/MyUDF.jar'

报以下错误：

```
Error: Error while compiling statement: FAILED: HiveAccessControlException
Permission denied: Principal [name=admin, type=USER] does not have following
privileges for operation CREATEFUNCTION [[ADMIN PRIVILEGE] on Object [type=DATABASE,
name=default], [ADMIN PRIVILEGE] on Object [type=FUNCTION, name=default.fn_test3]]
(state=42000,code=40000)
```

原因分析

Hive 中创建永久函数需要特殊的 role admin。

解决方案

在执行语句前执行 **set role admin 命令**即可解决。

10.17 Hive 服务健康状态和 Hive 实例健康状态的区别

问题现象

Hive 服务健康状态和 Hive 实例健康状态的区别是什么？

解决方案

Hive 服务的健康状态（也就是在 services 界面看到的健康状态）有 Good，Bad，Partially Healthy，Unknown 四种状态 ，四种状态除了取决于 Hive 本身服务的可用性（会用简单的 sql 来检测 Hive 服务的可用性），还取决于 Hive 服务所依赖的其他组件的服务状态。

Hive 实例分为 Hiveserver 和 Metastore 两种，健康状态有 Good，Concerning ，Unknown 三种状态，这三种状态是通过 jmx 通信来判定，与实例通信正常时为 Good，通信异常时为 Concerning，无法通信时为 Unknown。

10.18 Hive 中的告警有哪些以及触发的场景

Hive 中的告警

告警 ID	告警级别	可自动清除	告警名称	告警类型

告警 ID	告警级别	可自动清除	告警名称	告警类型
16000	Minor	TRUE	Percentage of Sessions Connected to the HiveServer to Maximum Number Allowed Exceeds the Threshold	故障告警
16001	Minor	TRUE	Hive Warehouse Space Usage Exceeds the Threshold	故障告警
16002	Minor	TRUE	The Successful Hive SQL Operations Lower than The Threshold	故障告警
16004	Critical	TRUE	Hive Service Unavailable	故障告警

告警触发场景

- 16000：当连接 HiveServer 的 session 数占允许连接总数的比率超过设定的阈值的时候触发告警。如连接的 session 数为 9，总连接数为 12，设定的阈值为 70%，9/12 > 70%便触发告警。
- 16001：当 Hive 使用的 HDFS 容量占分配给 Hive 的 HDFS 总容量的比率超过设定的阈值时触发告警。如分配给 Hive 的是 500G，Hive 已经使用 400G，设定的阈值时 75%，400/500 > 75%便触发告警。
- 16002：当执行 SQL 的成功率低于设定的阈值时变触发告警。如你执行了 4 条失败了 2 条，设定的阈值为 60%，成功率 2/4 < 60%便触发告警。
- 16004：Hive 服务的健康状态变为 Bad 时触发告警。

说明

- MRS Manager 界面操作：告警的阈值和告警的级别以及触发告警的时间段可以在 MRS Manager 界面的“系统设置 > 阈值配置”中设定。FusionInsight Manager 界面操作：告警的阈值和告警的级别以及触发告警的时间段可以在 FusionInsight Manager 界面的“运维 > 告警 > 阈值设置”中设定。
- Hive 运行相关的指标可以在 Hive 监控界面查看。

10.19 Shell 客户端连接提示"authentication failed"

问题现象

安全集群中，HiveServer 服务正常的情况下，Shell 客户端中执行 beeline 命令失败，界面提示“authentication failed”，如下：

```
Debug is true storeKey false useTicketCache true useKeyTab false doNotPrompt false
ticketCache is null isInitiator true KeyTab is null refreshKrb5Config is false
principal is null tryFirstPass is false useFirstPass is false storePass is false
clearPass is false
Acquire TGT from Cache
Credentials are no longer valid
Principal is null
null credentials from Ticket Cache
[Krb5LoginModule] authentication failed
No password provided
```

可能原因

- 客户端用户没有进行安全认证
- kerberos 认证超期

解决方案

步骤 1 登录 Hive 客户端所在节点。

步骤 2 执行 **source** *集群客户端安装目录*/**bigdata_env** 命令。

可通过 **klist** 命令查看本地是否有有效票据，如下信息表明票据在 16 年 12 月 24 日 14:11:42 生效，将在 16 年 12 月 25 日 14:11:40 失效。在此期间可以使用该票据，其他时间则该票据无效。

```
klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: xxx@HADOOP.COM
Valid starting    Expires          Service principal
12/24/16 14:11:42  12/25/16 14:11:40  krbtgt/HADOOP.COM@HADOOP.COM
```

步骤 3 执行 **kinit** *username* 进行认证，然后再使用客户端。

----结束

10.20 客户端提示访问 ZooKeeper 失败

问题现象

安全集群中，HiveServer 服务正常的情况下，通过 jdbc 接口连接 HiveServer 执行 sql 时报出 ZooKeeper 认证异常"The ZooKeeper client is AuthFailed"，如下：

```
14/05/19 10:52:00 WARN utils.HAClientUtilDummyWatcher: The ZooKeeper client is
AuthFailed
 14/05/19 10:52:00 INFO utils.HiveHAClientUtil: Exception thrown while reading data
from znode.The possible reason may be connectionless. This is recoverable.
Retrying..
 14/05/19 10:52:16 WARN utils.HAClientUtilDummyWatcher: The ZooKeeper client is
AuthFailed
 14/05/19 10:52:32 WARN utils.HAClientUtilDummyWatcher: The ZooKeeper client is
AuthFailed
 14/05/19 10:52:32 ERROR st.BasicTestCase: Exception: Could not establish
connection to active hiveserver
 java.sql.SQLException: Could not establish connection to active hiveserver
```

或者报出无法读取"Hiveserver2 configs from ZooKeeper"，如下：

```
Exception in thread "main" java.sql.SQLException:
org.apache.hive.jdbc.ZooKeeperHiveClientException: Unable to read HiveServer2
configs from ZooKeeper
at org.apache.hive.jdbc.HiveConnection.<init>(HiveConnection.java:144)
at org.apache.hive.jdbc.HiveDriver.connect(HiveDriver.java:105)
at java.sql.DriverManager.getConnection(DriverManager.java:664)
at java.sql.DriverManager.getConnection(DriverManager.java:247)
at JDBCExample.main(JDBCExample.java:82)
Caused by: org.apache.hive.jdbc.ZooKeeperHiveClientException: Unable to read
HiveServer2 configs from ZooKeeper
at
org.apache.hive.jdbc.ZooKeeperHiveClientHelper.configureConnParams(ZooKeeperHiveCli
entHelper.java:100)
at org.apache.hive.jdbc.Utils.configureConnParams(Utils.java:509)
at org.apache.hive.jdbc.Utils.parseURL(Utils.java:429)
at org.apache.hive.jdbc.HiveConnection.<init>(HiveConnection.java:142)
... 4 more
Caused by: org.apache.zookeeper.KeeperException$ConnectionLossException:
KeeperErrorCode = ConnectionLoss for /hiveserver2
at org.apache.zookeeper.KeeperException.create(KeeperException.java:99)
at org.apache.zookeeper.KeeperException.create(KeeperException.java:51)
at org.apache.zookeeper.ZooKeeper.getChildren(ZooKeeper.java:2374)
at
org.apache.curator.framework.imps.GetChildrenBuilderImpl$3.call(GetChildrenBuilderI
mpl.java:214)
at
org.apache.curator.framework.imps.GetChildrenBuilderImpl$3.call(GetChildrenBuilderI
mpl.java:203)
at org.apache.curator.RetryLo，op.callWithRetry(RetryLoop.java:107)
at
org.apache.curator.framework.imps.GetChildrenBuilderImpl.pathInForeground(GetChildr
enBuilderImpl.java:200)
at
org.apache.curator.framework.imps.GetChildrenBuilderImpl.forPath(GetChildrenBuilder
Impl.java:191)
at
org.apache.curator.framework.imps.GetChildrenBuilderImpl.forPath(GetChildrenBuilder
Impl.java:38)
```

可能原因

- 客户端连接 HiveServer 时，HiveServer 的地址是从 ZooKeeper 中自动获取，当 ZooKeeper 连接认证异常时，无法从 ZooKeeper 中获取正确的 HiveServer 地址。
- 在连接 zookeeper 认证时，需要客户端传入 krb5.conf，principal，keytab 等相关信息。认证失败有如下几种：
  - user.keytab 路径写错。
  - user.principal 写错。
  - 集群做过切换域名操作但客户端拼接 url 时使用旧的 principal。
  - 有防火墙相关设置，导致客户端本身无法通过 kerberos 认证，Kerberos 需要开放的端口有 21730(TCP)、21731(TCP/UDP)、21732(TCP/UDP)。

解决方案

步骤 1 确保用户可以正常读取客户端节点相关路径下的 user.keytab 文件。

步骤 2 确保用户的 user.principal 与指定的 keytab 文件对应。

可通过 **klist -kt keytabpath/user.keytab** 查看。

步骤 3 如果集群有做过切换域名操作，需要保证 url 中使用的 principal 字段是新域名。

如默认为 hive/hadoop.hadoop.com@HADOOP.COM，当集群有切换域名的操作时，该字段需要进行相关修改。如域名为 abc.com 时，则此处应填写 hive/hadoop.abc.com@ABC.COM。

步骤 4 确保可以正常的认证连接 HiveServer。

在客户端执行以下命令

source *客户端安装目录***/bigdata_env**

kinit *username*

然后再使用客户端执行 **beeline**，确保可以正常运行。

----结束

10.21 使用 udf 函数提示"Invalid function"

问题现象

在 Hive 客户端中使用 Spark 创建 UDF 函数时，报出"ERROR 10011","invalid function" 的异常，如下：

```
Error: Error while compiling statement: FAILED: SemanticException [Error 10011]:
Line 1:7 Invalid function 'test_udf' (state=42000,code=10011)
```

在多个 HiveServer 之间使用 UDF 也存在上述问题。例如，在 HiveServer1 中使用 HiverServer2 创建的 UDF，如果不及时同步元数据信息，连接 HiveServer1 的客户端也会提示上述错误信息。

可能原因

多个 HiveServer 之间或者 Hive 与 Spark 之间共用的元数据未同步，导致不同 HiveServer 实例内存数据不一致，造成 UDF 不生效。

解决方案

需要将新建的 UDF 信息同步到 HiveServer 中，执行 reload function 操作即可。

10.22 Hive 服务状态为 Unknown 总结

可能原因

Hive 服务停止。

解决方案

重启 Hive 服务。

10.23 Hiveserver 或者 Metastore 实例的健康状态为 unknown

问题现象

hiveserver 或者 metastore 实例的健康状态为 unknown。

可能原因

hiveserver 或者 metastore 实例被停止。

解决方案

重启 hiveserver 或者 metastore 实例。

10.24 Hiveserver 或者 Metastore 实例的健康状态为 Concerning

问题现象

Hiveserver 或者 Metastore 实例的健康状态为 Concerning。

可能原因

hiveserver 或者 metastore 实例在启动的时候发生异常，无法正常启动。如，当修改 MetaStore/HiveServer GC 参数时，可通过查看对应进程的启动日志，如 hiveserver.out(hadoop-omm-jar-192-168-1-18.out)文件排查。 如下异常：

Error: Could not find or load main class Xmx2048M

说明 java 虚拟机启动时，将 Xmx2048M 作为 java 进程的启动参数而不是 JVM 的启动参数了，如下将符号‘-’误删掉。

```
METASTORE_GC_OPTS=Xms1024M Xmx2048M -DIgnoreReplayReqDetect
-XX\:CMSFullGCsBeforeCompaction\=1 -XX\:+UseConcMarkSweepGC
-XX\:+CMSParallelRemarkEnabled -XX\:+UseCMSCompactAtFullCollection
-XX\:+ExplicitGCInvokesConcurrent -server -XX\:MetaspaceSize\=128M
-XX\:MaxMetaspaceSize\=256M
```

解决方案

因此遇到此类异常应该检查最近的变更项，以确认是否配置有误。

```
METASTORE_GC_OPTS=Xms1024M -Xmx2048M -DIgnoreReplayReqDetect
-XX\:CMSFullGCsBeforeCompaction\=1 -XX\:+UseConcMarkSweepGC
-XX\:+CMSParallelRemarkEnabled -XX\:+UseCMSCompactAtFullCollection
-XX\:+ExplicitGCInvokesConcurrent -server -XX\:MetaspaceSize\=128M
-XX\:MaxMetaspaceSize\=256M
```

10.25 TEXTFILE 类型文件使用 ARC4 压缩时 select 结果乱码

问题现象

Hive 查询结果表做压缩存储（ARC4），对结果表做 select * 查询时返回结果为乱码。

可能原因

Hive 默认压缩格式不是 ARC4 格式或者未开启输出压缩。

解决方案

步骤 1 在 select 结果乱码时，在 beeline 中进行如下设置。

set mapreduce.output.fileoutputformat.compress.codec=org.apache.hadoop.io.encryption.arc4.ARC4BlockCodec;

set hive.exec.compress.output=true;

步骤 2 使用块解压的方式先将表导入一个新表中。

insert overwrite table tbl_result select * from tbl_source;

步骤 3 再进行查询。

select * from tbl_result;

----结束

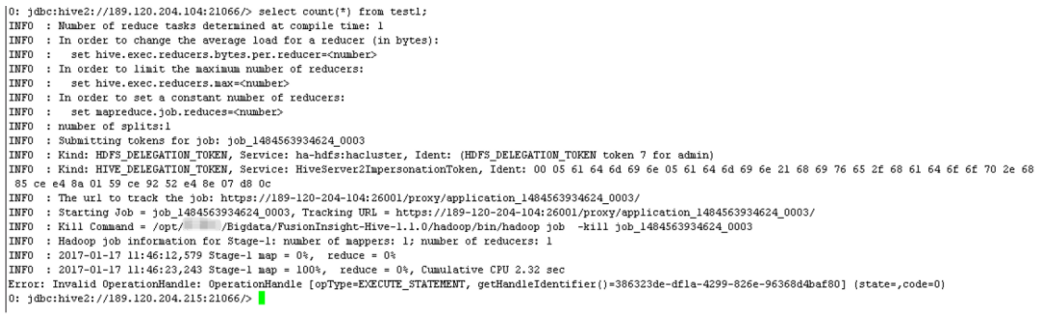
10.26 hive 任务运行过程中失败，重试成功

问题现象

当 hive 任务在正常运行时失败，在客户端报出错误，类似的错误打印：

```
Error:Invalid OperationHandler:OperationHander
[opType=EXECUTE_STATEMENT,getHandleIdentifier()=XXX](state=,code=0)
```

而此任务提交到 yarn 上的 mapreduce 任务运行成功。



原因分析

出错的集群有两个 hiveserver 实例，首先查看其中一个 hiveserver 日志发现里面的报错与客户端中的错误一样均是 Error:Invalid OperationHandler，查看另一个 hiveserver 发现在出错的时间段此实例有如下类似 START_UP 的打印，说明那段时间进程被停止过，后来又启动成功，提交的任务本来连接的是重启过的 hiveserver 实例，当这个实例被停止后，任务进程连接到另一个健康的 hiveserver 上导致报错。

```
2017-02-15 14:40:11,309 | INFO  | main | STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting HiveServer2
STARTUP_MSG:   host = XXX-120-85-154/XXX.120.85.154
STARTUP_MSG:   args = []
STARTUP_MSG:   version = 1.3.0
```

解决办法

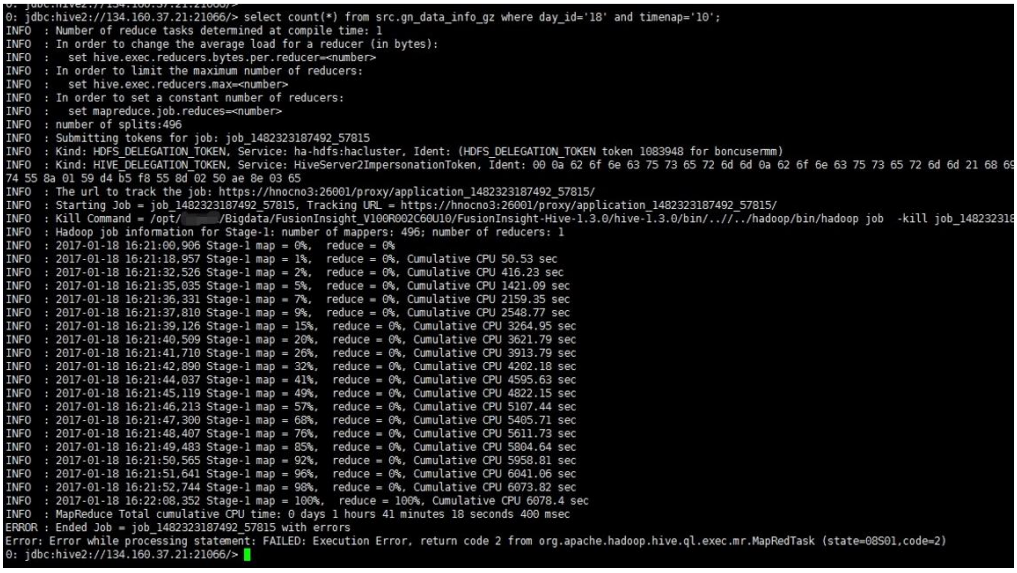
重新提交一次任务即可，保证在任务执行期间不手动重启 hiveserver 进程。

10.27 执行 select 语句报错

问题现象

执行语句 select count(*) from XXX;时客户端报错：Error:Error while processing statement :FAILED:Execution Error,return code 2 from ...

这个报错 return code2 说明是在执行 mapreduce 任务期间报错导致任务失败。

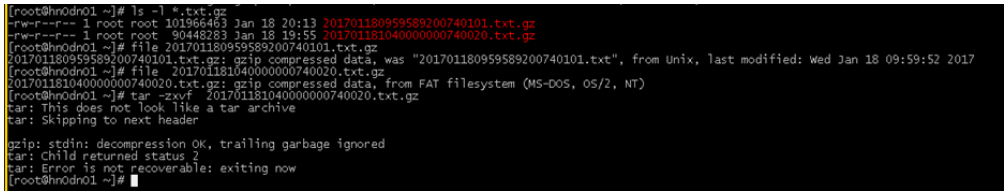


原因分析

1. 进入 yarn 原生页面查看 mapreduce 任务的日志看到报错是无法识别到压缩方式导致错误，看文件后缀是 gzip 压缩，堆栈却报出是 zlib 方式。

```
2017-01-18 16:22:07,566 INFO [main] org.apache.hadoop.hive.ql.exec.MapOperator: 4 Close done
2017-01-18 16:22:07,572 WARN [main] org.apache.hadoop.mapred.YarnChild: Exception running child : java.io.IOException: java.io.IOException: unknown compression method
        at org.apache.hadoop.hive.io.HiveIOExceptionHandlerChain.handleRecordReaderNextException(HiveIOExceptionHandlerChain.java:121)
        at org.apache.hadoop.hive.io.HiveIOExceptionHandlerUtil.handleRecordReaderNextException(HiveIOExceptionHandlerUtil.java:77)
        at org.apache.hadoop.hive.ql.io.HiveContextAwareRecordReader.doNext(HiveContextAwareRecordReader.java:355)
        at org.apache.hadoop.hive.ql.io.HiveRecordReader.doNext(HiveRecordReader.java:79)
        at org.apache.hadoop.hive.ql.io.HiveRecordReader.doNext(HiveRecordReader.java:33)
        at org.apache.hadoop.hive.ql.io.HiveContextAwareRecordReader.next(HiveContextAwareRecordReader.java:116)
        at org.apache.hadoop.mapred.MapTask$TrackedRecordReader.moveToNext(MapTask.java:199)
        at org.apache.hadoop.mapred.MapTask$TrackedRecordReader.next(MapTask.java:185)
        at org.apache.hadoop.mapred.MapRunner.run(MapRunner.java:52)
        at org.apache.hadoop.mapred.MapTask.runOldMapper(MapTask.java:453)
        at org.apache.hadoop.mapred.MapTask.run(MapTask.java:343)
        at org.apache.hadoop.mapred.YarnChild$2.run(YarnChild.java:180)
        at java.security.AccessController.doPrivileged(Native Method)
        at javax.security.auth.Subject.doAs(Subject.java:422)
        at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1726)
        at org.apache.hadoop.mapred.YarnChild.main(YarnChild.java:174)
Caused by: java.io.IOException: unknown compression method
        at org.apache.hadoop.io.compress.zlib.ZlibDecompressor.inflateBytesDirect(Native Method)
        at org.apache.hadoop.io.compress.zlib.ZlibDecompressor.decompress(ZlibDecompressor.java:225)
        at org.apache.hadoop.io.compress.DecompressorStream.decompress(DecompressorStream.java:91)
        at org.apache.hadoop.io.compress.DecompressorStream.read(DecompressorStream.java:85)
        at java.io.InputStream.read(InputStream.java:101)
        at org.apache.hadoop.util.LineReader.fillBuffer(LineReader.java:180)
        at org.apache.hadoop.util.LineReader.readDefaultLine(LineReader.java:216)
        at org.apache.hadoop.util.LineReader.readLine(LineReader.java:174)
        at org.apache.hadoop.mapred.LineRecordReader.next(LineRecordReader.java:248)
        at org.apache.hadoop.mapred.LineRecordReader.next(LineRecordReader.java:48)
        at org.apache.hadoop.hive.ql.io.HiveContextAwareRecordReader.doNext(HiveContextAwareRecordReader.java:350)
        ... 13 more

2017-01-18 16:22:07,576 INFO [main] org.apache.hadoop.mapred.Task: Runnning cleanup for the task
```

2. 因此怀疑此语句查询的表对应的 HDFS 上的文件有问题，map 日志中打印出了解析的对应的文件名，将其从 HDFS 上下载到本地，看到是 gz 结尾的文件，使用 **tar** 命令解压报错，格式不正确无法解压。使用 file 命令查看文件属性发现此文件来自于 FAT 系统的压缩而非 unix。



解决办法

将格式不正确的文件移除 hdfs 目录或者替换为正确的格式的文件。

10.28 drop partition 操作，有大量分区时操作失败

问题背景与现象

执行 drop　partitions 操作，执行异常：

```
MetaStoreClient lost connection. Attempting to reconnect. |
org.apache.hadoop.hive.metastore.RetryingMetaStoreClient.invoke(RetryingMetaStoreCl
ient.java:187)
org.apache.thrift.transport.TTransportException
at org.apache.thrift.transport.TIOStreamTransport.read(TIOStreamTransport.java:132)
at org.apache.thrift.transport.TTransport.xxx(TTransport.java:86)
at org.apache.thrift.transport.TSaslTransport.readLength(TSaslTransport.java:376)
at org.apache.thrift.transport.TSaslTransport.readFrame(TSaslTransport.java:453)
at org.apache.thrift.transport.TSaslTransport.read(TSaslTransport.java:435)
...
```

查看对应 metaStore 日志，有 StackOverFlow 异常

```
2017-04-22 01:00:58,834 | ERROR | pool-6-thread-208 | java.lang.StackOverflowError
at org.datanucleus.store.rdbms.sql.SQLText.toSQL(SQLText.java:330)
at org.datanucleus.store.rdbms.sql.SQLText.toSQL(SQLText.java:339)
at org.datanucleus.store.rdbms.sql.SQLText.toSQL(SQLText.java:339)
at org.datanucleus.store.rdbms.sql.SQLText.toSQL(SQLText.java:339)
at org.datanucleus.store.rdbms.sql.SQLText.toSQL(SQLText.java:339)
```

原因分析

drop partition 的处理逻辑是将找到所有满足条件的分区，将其拼接起来，最后统一删除。由于分区数过多，拼删元数据堆栈较深，出现 StackOverFlow 异常。

解决办法

分批次删除分区。

10.29 localtask 启动失败

问题背景与现象

1. 执行 join 等操作，数据量较小时，会启动 localtask 执行，执行过程会报错：

```
jdbc:hive2://10.*.*.*:21066/> select a.name ,b.sex from student a join student1
b on (a.name = b.name);
ERROR : Execution failed with exit status: 1
ERROR : Obtaining error information
ERROR :
Task failed!
Task ID:
  Stage-4
...
Error: Error while processing statement: FAILED: Execution Error, return code 1
from org.apache.hadoop.hive.ql.exec.mr.MapredLocalTask (state=08S01,code=1)
...
```

2. 查看对应 hiveserver 日志，发现是启动 localtask 失败

```
2018-04-25 16:37:19,296 | ERROR | HiveServer2-Background-Pool: Thread-79 |
Execution failed with exit status: 1 |
org.apache.hadoop.hive.ql.session.SessionState$LogHelper.printError(SessionStat
e.java:1016)
2018-04-25 16:37:19,296 | ERROR | HiveServer2-Background-Pool: Thread-79 |
Obtaining error information |
org.apache.hadoop.hive.ql.session.SessionState$LogHelper.printError(SessionStat
e.java:1016)
2018-04-25 16:37:19,297 | ERROR | HiveServer2-Background-Pool: Thread-79 |
Task failed!
Task ID:
  Stage-4
Logs:
 |
org.apache.hadoop.hive.ql.session.SessionState$LogHelper.printError(SessionStat
e.java:1016)
2018-04-25 16:37:19,297 | ERROR | HiveServer2-Background-Pool: Thread-79 |
/var/log/Bigdata/hive/hiveserver/hive.log |
org.apache.hadoop.hive.ql.session.SessionState$LogHelper.printError(SessionStat
e.java:1016)
2018-04-25 16:37:19,297 | ERROR | HiveServer2-Background-Pool: Thread-79 |
Execution failed with exit status: 1 |
org.apache.hadoop.hive.ql.exec.mr.MapredLocalTask.executeInChildVM(MapredLocalT
ask.java:342)
2018-04-25 16:37:19,309 | ERROR | HiveServer2-Background-Pool: Thread-79 |
FAILED: Execution Error, return code 1 from
org.apache.hadoop.hive.ql.exec.mr.MapredLocalTask |
org.apache.hadoop.hive.ql.session.SessionState$LogHelper.printError(SessionStat
e.java:1016)
...
2018-04-25 16:37:36,438 | ERROR | HiveServer2-Background-Pool: Thread-88 |
Error running hive query:  |
org.apache.hive.service.cli.operation.SQLOperation$1$1.run(SQLOperation.java:24
8)
org.apache.hive.service.cli.HiveSQLException: Error while processing statement:
```

```
FAILED: Execution Error, return code 1 from
org.apache.hadoop.hive.ql.exec.mr.MapredLocalTask
       at
org.apache.hive.service.cli.operation.Operation.toSQLException(Operation.java:3
39)
       at
org.apache.hive.service.cli.operation.SQLOperation.runQuery(SQLOperation.java:1
69)
       at
org.apache.hive.service.cli.operation.SQLOperation.access$200(SQLOperation.java
:75)
       at
org.apache.hive.service.cli.operation.SQLOperation$1$1.run(SQLOperation.java:24
5)
       at java.security.AccessController.doPrivileged(Native Method)
       at javax.security.auth.Subject.doAs(Subject.java:422)
       at
org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:
1710)
       at
org.apache.hive.service.cli.operation.SQLOperation$1.run(SQLOperation.java:258)
       at
java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:511)
       at java.util.concurrent.FutureTask.run(FutureTask.java:266)
       at
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
       at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
       at java.lang.Thread.run(Thread.java:745)
```

3. 查看对应 hiveserver 日志目录(/var/log/Bigdata/hive/hiveserver)下，hs_err_pid_*****.log，发现有内存不够的错误

```
# There is insufficient memory for the Java Runtime Environment to continue.
# Native memory allocation (mmap) failed to map 20776943616 bytes for
committing reserved memory.
  ...
```

原因分析

Hive 在执行 join 操作，数据量小时会生成 MapJoin，执行 MapJoin 时会生成 localtask 任务，localtask 启动的 jvm 内存继承了父进程的内存。

当有多个 join 执行的时候，启动多个 localtask，如果机器内存不够，就会导致启动 localtask 失败。

解决办法

步骤 1 搜索“hive.auto.convert.join”参数并修改 hive 的配置 hive.auto.convert.join 为 false，保存配置并重启服务。

该参数修改后会对业务性能有一定影响。继续执行后续步骤可不影响业务性能。

步骤 2 搜索“HIVE_GC_OPTS”参数并修改 hive 的 HIVE_GC_OPTS，把 Xms 调小，具体要根据业务评估，最小设置为 Xmx 的一半，修改完后保存配置并重启服务。

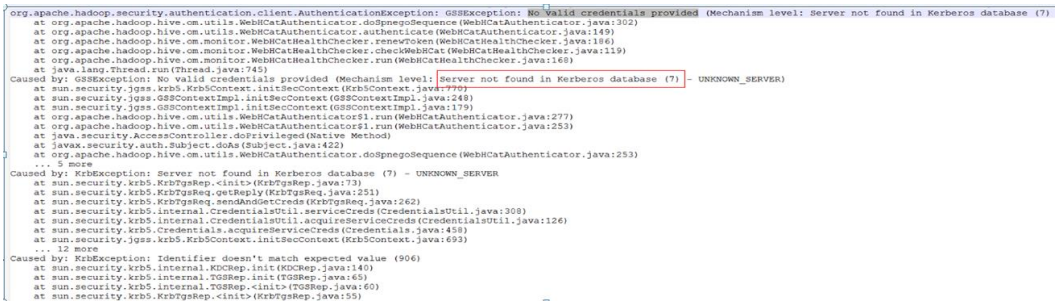
----结束

10.30 WebHCat 启动失败

问题背景与现象

用户修改 hostname 导致 WebHCat 启动失败。

查看对应节点 WebHCat 启动日志（ /var/log/Bigdata/hive/webhcat/hive.log），发现报如下错误：



原因分析

1. MRSwebhcat 角色的服务端账户中涉及到 hostname，如果安装完后再修改 hostname，就会导致启动失败。
2. /etc/hosts 中配置了一对多或者多对一的主机名和 IP 对应关系，导致在执行 **hostname** 和 **hostname -i** 获取不到正确的 IP 和 hostname。

解决办法

步骤 1 将修改了节点的 hostname 全部修改为集群安装前的 hostname。

步骤 2 排查 WebHCat 所在节点的/etc/hosts 是否配置正确。

步骤 3 重启 WebHCat。

----结束

10.31 切域后 Hive 二次开发样例代码报错

问题背景与现象

hive 的二次开发代码样例运行报 No rules applied to ****的错误：

原因分析

1. hive 的二次开发样例代码会加载 core-site.xml，此文件默认是通过 classload 加载，所以使用的时候要把此配置文件放到启动程序的 classpath 路径下面。
2. 如果修改了集群的域名，那么 core-site.xml 将发生变化，需要下载最新的 core-site.xml 并放入到打包 hive 二次开发样例代码进程的 classpath 路径下面。

解决办法

步骤 1 下载集群 Hive 最新的客户端，获取最新的 core-site.xml。

步骤 2 将 core-site.xml 放入到打包 hive 二次开发样例代码进程的 classpath 路径下面。

----结束

10.32 DBService 超过最大连接数，导致 metastore 异常

问题背景与现象

DBService 默认最大连接数是 300，如果当业务量比较大，导致连接 DBService 的最大连接数超过 300 时，metastore 会出现异常，并报 slots are reserved for non-replication superuser connections 的错误：

```
2018-04-26 14:58:55,657 | ERROR | BoneCP-pool-watch-thread | Failed to acquire
connection to jdbc:postgresql://10.*.*.*:20051/hivemeta?socketTimeout=60. Sleeping
for 1000 ms. Attempts left: 9 |
com.jolbox.bonecp.BoneCP.obtainInternalConnection(BoneCP.java:292)
org.postgresql.util.PSQLException: FATAL: remaining connection slots are reserved
for non-replication superuser connections
    at
org.postgresql.core.v3.ConnectionFactoryImpl.readStartupMessages(ConnectionFactoryI
mpl.java:643)
    at
org.postgresql.core.v3.ConnectionFactoryImpl.openConnectionImpl(ConnectionFactoryIm
pl.java:184)
    at
org.postgresql.core.ConnectionFactory.openConnection(ConnectionFactory.java:64)
    at
org.postgresql.jdbc2.AbstractJdbc2Connection.<init>(AbstractJdbc2Connection.java:12
```

```
4)
    at
org.postgresql.jdbc3.AbstractJdbc3Connection.<init>(AbstractJdbc3Connection.java:28)
    at
org.postgresql.jdbc3g.AbstractJdbc3gConnection.<init>(AbstractJdbc3gConnection.java
:20)
    at
org.postgresql.jdbc4.AbstractJdbc4Connection.<init>(AbstractJdbc4Connection.java:30)
    at org.postgresql.jdbc4.Jdbc4Connection.<init>(Jdbc4Connection.java:22)
    at org.postgresql.Driver.makeConnection(Driver.java:392)
    at org.postgresql.Driver.connect(Driver.java:266)
    at java.sql.DriverManager.getConnection(DriverManager.java:664)
    at java.sql.DriverManager.getConnection(DriverManager.java:208)
    at com.jolbox.bonecp.BoneCP.obtainRawInternalConnection(BoneCP.java:361)
    at com.jolbox.bonecp.BoneCP.obtainInternalConnection(BoneCP.java:269)
    at com.jolbox.bonecp.ConnectionHandle.<init>(ConnectionHandle.java:242)
    at com.jolbox.bonecp.PoolWatchThread.fillConnections(PoolWatchThread.java:115)
    at com.jolbox.bonecp.PoolWatchThread.run(PoolWatchThread.java:82)
    at
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
    at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
    at java.lang.Thread.run(Thread.java:745)
```

原因分析

业务量大导致连接 DBService 的最大连接数超过了 300，需要修改 DBService 的最大连接数。

解决办法

步骤 1 搜索 dbservice.database.max.connections 配置项，并修改 dbservice.database.max.connections 配置的值到合适值，不能超过 1000。

步骤 2 保存配置，并重启受影响的服务或者实例。

步骤 3 如果调整完还报超过最大连接数，需要排查业务代码，是否有连接泄露。

----结束

10.33 beeline 报 Failed to execute session hooks: over max connections 错误

问题背景与现象

HiveServer 连接的最大连接数默认为 200，当超过 200 时，beeline 会报 Failed to execute session hooks: over max connections

```
beeline> [root@172-27-16-38 c70client]# beeline
Connecting to
jdbc:hive2://129.188.82.38:24002,129.188.82.36:24002,129.188.82.35:24002/;serviceDi
```

```
scoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2;sasl.qop=auth-
conf;auth=KERBEROS;principal=hive/hadoop.hadoop.com@HADOOP.COM
Debug is  true storeKey false useTicketCache true useKeyTab false doNotPrompt false
ticketCache is null isInitiator true KeyTab is null refreshKrb5Config is false
principal is null tryFirstPass is false useFirstPass is false storePass is false
clearPass is false
Acquire TGT from Cache
Principal is xxx@HADOOP.COM
Commit Succeeded

Error: Failed to execute session hooks: over max connections. (state=,code=0)
Beeline version 1.2.1 by Apache Hive
```

查看 hiveserver 日志(/var/log/Bigdata/hive/hiveserver/hive.log)报 over max connections 错误

```
2018-05-03 04:31:56,728 | WARN  | HiveServer2-Handler-Pool: Thread-137 | Error
opening session:  |
org.apache.hive.service.cli.thrift.ThriftCLIService.OpenSession(ThriftCLIService.ja
va:542)
org.apache.hive.service.cli.HiveSQLException: Failed to execute session hooks: over
max connections.
    at
org.apache.hive.service.cli.session.SessionManager.openSession(SessionManager.java:
322)
    at
org.apache.hive.service.cli.CLIService.openSessionWithImpersonation(CLIService.java
:189)
    at
org.apache.hive.service.cli.thrift.ThriftCLIService.getSessionHandle(ThriftCLIServi
ce.java:663)
    at
org.apache.hive.service.cli.thrift.ThriftCLIService.OpenSession(ThriftCLIService.ja
va:527)
    at
org.apache.hive.service.cli.thrift.TCLIService$Processor$OpenSession.getResult(TCLI
Service.java:1257)
    at
org.apache.hive.service.cli.thrift.TCLIService$Processor$OpenSession.getResult(TCLI
Service.java:1242)
    at org.apache.thrift.ProcessFunction.process(ProcessFunction.java:39)
    at org.apache.thrift.TBaseProcessor.process(TBaseProcessor.java:39)
    at
org.apache.hadoop.hive.thrift.HadoopThriftAuthBridge$Server$TUGIAssumingProcessor.p
rocess(HadoopThriftAuthBridge.java:710)
    at
org.apache.thrift.server.TThreadPoolServer$WorkerProcess.run(TThreadPoolServer.java
:286)
    at
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
    at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
    at java.lang.Thread.run(Thread.java:745)
Caused by: org.apache.hive.service.cli.HiveSQLException: over max connections.
    at
org.apache.hadoop.hive.transporthook.SessionControllerTsaslTransportHook.checkTotal
```

```
SessionNumber(SessionControllerTsaslTransportHook.java:208)
    at
org.apache.hadoop.hive.transporthook.SessionControllerTsaslTransportHook.postOpen(S
essionControllerTsaslTransportHook.java:163)
    at
org.apache.hadoop.hive.transporthook.SessionControllerTsaslTransportHook.run(Sessio
nControllerTsaslTransportHook.java:134)
    at
org.apache.hive.service.cli.session.SessionManager.executeSessionHooks(SessionManag
er.java:432)
    at
org.apache.hive.service.cli.session.SessionManager.openSession(SessionManager.java:
314)
    ... 12 more
```

原因分析

业务量大导致连接 HiveServer 单个节点最大连接数超过了 200，需要调大连接 HiveServer 实例的最大连接数。

解决办法

步骤 1 搜索 hive.server.session.control.maxconnections 配置项，并修改 hive.server.session.control.maxconnections 配置的值到合适值，不能超过 1000。

步骤 2 保存配置并重启受影响的服务或者实例。

----结束

10.34 beeline 报 OutOfMemoryError 错误

问题背景与现象

beeline 客户端查询大量数据时，报 OutOFMemoryError：Java heap space，具体报错信息如下：

```
org.apache.thrift.TException: Error in calling method FetchResults
    at
org.apache.hive.jdbc.HiveConnection$SynchronizedHandler.invoke(HiveConnection.java:
1514)
    at com.sun.proxy.$Proxy4.FetchResults(Unknown Source)
    at org.apache.hive.jdbc.HiveQueryResultSet.next(HiveQueryResultSet.java:358)
    at org.apache.hive.beeline.BufferedRows.<init>(BufferedRows.java:42)
    at org.apache.hive.beeline.BeeLine.print(BeeLine.java:1856)
    at org.apache.hive.beeline.Commands.execute(Commands.java:873)
    at org.apache.hive.beeline.Commands.sql(Commands.java:714)
    at org.apache.hive.beeline.BeeLine.dispatch(BeeLine.java:1035)
    at org.apache.hive.beeline.BeeLine.execute(BeeLine.java:821)
    at org.apache.hive.beeline.BeeLine.begin(BeeLine.java:778)
    at org.apache.hive.beeline.BeeLine.mainWithInputRedirection(BeeLine.java:486)
    at org.apache.hive.beeline.BeeLine.main(BeeLine.java:469)
Caused by: java.lang.OutOfMemoryError: Java heap space
```

```
    at com.sun.crypto.provider.CipherCore.doFinal(CipherCore.java:959)
    at com.sun.crypto.provider.CipherCore.doFinal(CipherCore.java:824)
    at com.sun.crypto.provider.AESCipher.engineDoFinal(AESCipher.java:436)
    at javax.crypto.Cipher.doFinal(Cipher.java:2223)
    at
sun.security.krb5.internal.crypto.dk.AesDkCrypto.decryptCTS(AesDkCrypto.java:414)
    at
sun.security.krb5.internal.crypto.dk.AesDkCrypto.decryptRaw(AesDkCrypto.java:291)
    at sun.security.krb5.internal.crypto.Aes256.decryptRaw(Aes256.java:86)
    at sun.security.jgss.krb5.CipherHelper.aes256Decrypt(CipherHelper.java:1397)
    at sun.security.jgss.krb5.CipherHelper.decryptData(CipherHelper.java:576)
    at sun.security.jgss.krb5.WrapToken_v2.getData(WrapToken_v2.java:130)
    at sun.security.jgss.krb5.WrapToken_v2.getData(WrapToken_v2.java:105)
    at sun.security.jgss.krb5.Krb5Context.unwrap(Krb5Context.java:1058)
    at sun.security.jgss.GSSContextImpl.unwrap(GSSContextImpl.java:403)
    at com.sun.security.sasl.gsskerb.GssKrb5Base.unwrap(GssKrb5Base.java:77)
    at
org.apache.thrift.transport.TSaslTransport$SaslParticipant.unwrap(TSaslTransport.ja
va:559)
   at  org.apache.thrift.transport.TSaslTransport.readFrame(TSaslTransport.java:462)
    at org.apache.thrift.transport.TSaslTransport.read(TSaslTransport.java:435)
    at
org.apache.thrift.transport.TSaslClientTransport.read(TSaslClientTransport.java:37)
    at org.apache.thrift.transport.TTransport.xxx(TTransport.java:86)
    at org.apache.hadoop.hive.thrift.TFilterTransport.xxx(TFilterTransport.java:62)
    at org.apache.thrift.protocol.TBinaryProtocol.xxx(TBinaryProtocol.java:429)
    at org.apache.thrift.protocol.TBinaryProtocol.readI32(TBinaryProtocol.java:318)
    at
org.apache.thrift.protocol.TBinaryProtocol.readMessageBegin(TBinaryProtocol.java:21
9)
    at org.apache.thrift.TServiceClient.receiveBase(TServiceClient.java:77)
    at
org.apache.hive.service.cli.thrift.TCLIService$Client.recv_FetchResults(TCLIService
.java:505)
    at
org.apache.hive.service.cli.thrift.TCLIService$Client.FetchResults(TCLIService.java
:492)
    at sun.reflect.GeneratedMethodAccessor2.invoke(Unknown Source)
    at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:4
3)
    at java.lang.reflect.Method.invoke(Method.java:498)
    at
org.apache.hive.jdbc.HiveConnection$SynchronizedHandler.invoke(HiveConnection.java:
1506)
    at com.sun.proxy.$Proxy4.FetchResults(Unknown Source)
    at org.apache.hive.jdbc.HiveQueryResultSet.next(HiveQueryResultSet.java:358)
Error: Error retrieving next row (state=,code=0)
```

原因分析

- 客户查询大量数据，数据量过大。
- 客户在检索数据时使用 **select * from table_name;**，进行全表查询，表内数据过多。
- beeline 默认启动内存 128M，查询时返回结果集过大，导致 beeline 无法承载导致。

解决办法

步骤 1 执行 **select count(*) from table_name;**前确认需要查询的数据量大小，确认是否需要在 beeline 中显示如此数量级的数据。

步骤 2 如数量在一定范围内需要显示，请调整 hive 客户端的 jvm 参数， 在 hive 客户端目录/Hive 下的 component_env 中添加 export HIVE_OPTS=-Xmx1024M(具体数值请根据业务调整)，并重新执行 **source** *客户端目录*/**bigdata_env** 配置环境变量。

----结束

10.35 输入文件数超出设置限制导致任务执行失败

问题背景与现象

Hive 执行查询操作时报 Job Submission failed with exception 'java.lang.RuntimeException(input file number exceeded the limits in the conf;input file num is: 2380435,max heap memory is: 16892035072,the limit conf is: 500000/4)'，此报错中具体数值根据实际情况会发生变化，具体报错信息如下:

```
ERROR : Job Submission failed with exception 'java.lang.RuntimeException(input file
numbers exceeded the limits in the conf;
 input file num is: 2380435 ,
 max heap memory is: 16892035072 ,
 the limit conf is: 500000/4)'
java.lang.RuntimeException: input file numbers exceeded the limits in the conf;
 input file num is: 2380435 ,
 max heap memory is: 16892035072 ,
 the limit conf is: 500000/4
    at
org.apache.hadoop.hive.ql.exec.mr.ExecDriver.checkFileNum(ExecDriver.java:545)
    at org.apache.hadoop.hive.ql.exec.mr.ExecDriver.execute(ExecDriver.java:430)
    at org.apache.hadoop.hive.ql.exec.mr.MapRedTask.execute(MapRedTask.java:137)
    at org.apache.hadoop.hive.ql.exec.Task.executeTask(Task.java:158)
    at org.apache.hadoop.hive.ql.exec.TaskRunner.runSequential(TaskRunner.java:101)
    at org.apache.hadoop.hive.ql.Driver.launchTask(Driver.java:1965)
    at org.apache.hadoop.hive.ql.Driver.execute(Driver.java:1723)
    at org.apache.hadoop.hive.ql.Driver.runInternal(Driver.java:1475)
    at org.apache.hadoop.hive.ql.Driver.run(Driver.java:1283)
    at org.apache.hadoop.hive.ql.Driver.run(Driver.java:1278)
    at
org.apache.hive.service.cli.operation.SQLOperation.runQuery(SQLOperation.java:167)
    at
org.apache.hive.service.cli.operation.SQLOperation.access$200(SQLOperation.java:75)
    at
org.apache.hive.service.cli.operation.SQLOperation$1$1.run(SQLOperation.java:245)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:422)
    at
org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1710)
    at
org.apache.hive.service.cli.operation.SQLOperation$1.run(SQLOperation.java:258)
```

```
    at java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:511)
    at java.util.concurrent.FutureTask.run(FutureTask.java:266)
    at
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
    at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
    at java.lang.Thread.run(Thread.java:745)

Error: Error while processing statement: FAILED: Execution Error, return code 1
from org.apache.hadoop.hive.ql.exec.mr.MapRedTask (state=08S01,code=1)
```

原因分析

MapReduce 任务提交前对输入文件数的检查策略：在提交的 MapReduce 任务中，允许的最大输入文件数和 HiveServer 最大堆内存的比值，例如 500000/4（默认值），表示每 4GB 堆内存最大允许 500000 个输入文件。在输入的文件数超出此限制时则会发生此错误。

解决办法

步骤 1 搜索 hive.mapreduce.input.files2memory 配置项，并修改 hive.mapreduce.input.files2memory 配置的值到合适值，根据实际内存和任务情况对此值进行调整。

步骤 2 保存配置并重启受影响的服务或者实例。

步骤 3 如调整后问题仍未解决，请根据业务情况调整 HiveServer 的 GC 参数至合理的值。

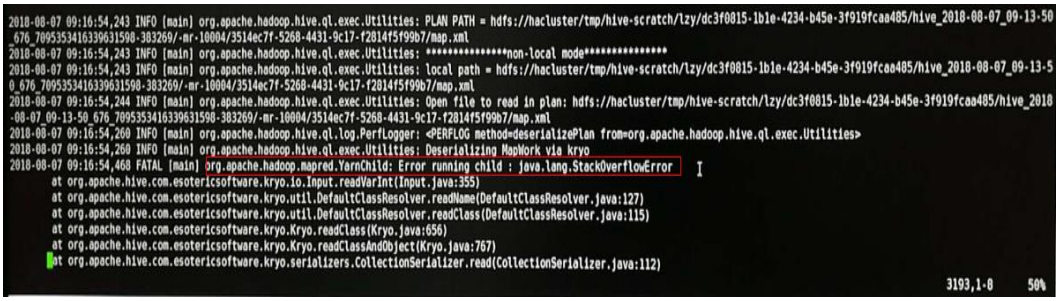
----结束

10.36 任务执行中报栈内存溢出导致任务执行失败

问题背景与现象

Hive 执行查询操作时报错 **Error running child : java.lang.StackOverflowError**，具体报错信息如下：

```
FATAL [main] org.apache.hadoop.mapred.YarnChild: Error running child :
java.lang.StackOverflowError
at org.apache.hive.come.esotericsoftware.kryo.io.Input.readVarInt(Input.java:355)
at
org.apache.hive.come.esotericsoftware.kryo.util.DefautClassResolver.readName(Defaut
ClassResolver.java:127)
at
org.apache.hive.come.esotericsoftware.kryo.util.DefautClassResolver.readClass(Defau
tClassResolver.java:115)
at org.apache.hive.come.esotericsoftware.kryo.Kryo.readClass(Kryo.java.656)
at  org.apache.hive.come.esotericsoftware.kryo.kryo.readClassAndObject(Kryo.java:767)
at
org.apache.hive.come.esotericsoftware.kryo.serializers.collectionSerializer.read(Co
llectionSerializer.java:112)
```



原因分析

java.lang.StackOverflowError 这是内存溢出错误的一种，即线程栈的溢出，方法调用层次过多（比如存在无限递归调用）或线程栈太小都会导致此报错。

解决办法

通过调整 mapreduce 阶段的 map 和 reduce 子进程 JVM 参数中的栈内存解决此问题，主要涉及参数为 mapreduce.map.java.opts（调整 map 的栈内存）和 mapreduce.reduce.java.opts（调整 reduce 的栈内存），调整方法如下（以 mapreduce.map.java.opts 参数为例）。

- 临时增加 map 内存（只针对此次 beeline 生效）：

  在 beeline 中执行如下命令 **set mapreduce.map.java.opts=-Xss8G;**（具体数值请结合实际业务情况进行调整）。

- 永久增加 map 内存 mapreduce.map.memory.mb 和 mapreduce.map.java.opts 的值：

  a. 在 hiveserver 自定义参数界面添加自定义参数 mapreduce.map.java.opts 及相应的值。

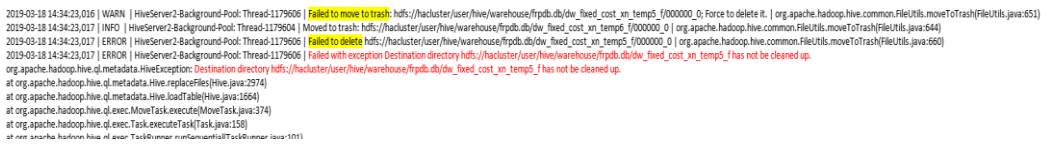
  b. 保存配置并重启受影响的服务或者实例。

  修改配置后需要保存，请注意参数在 HiveServer 自定义参数处修改，保存重启后生效（重启期间 hive 服务不可用），请注意执行时间窗口。

10.37 对同一张表或分区并发写数据导致任务失败

问题背景与现象

Hive 执行插入语句时，报错 HDFS 上文件或目录已存在或被清除，具体报错如下：



原因分析

1. 根据 HiveServer 的审计日志，确认该任务的开始时间和结束时间。
2. 在上述时间区间内，查找是否有对同一张表或分区进行插入数据的操作。

3. Hive 不支持对同一张表或分区进行并发数据插入，这样会导致多个任务操作同一个数据临时目录，一个任务将另一个任务的数据移走，导致任务失败。

解决办法

修改业务逻辑，单线程插入数据到同一张表或分区。

10.38 Hive 任务失败，报没有 HDFS 目录的权限

问题背景与现象

Hive 任务报错，提示执行用户没有 HDFS 目录权限

```
2019-04-09 17:49:19,845 | ERROR | HiveServer2-Background-Pool: Thread-3160445 | Job Submission failed with exception
'org.apache.hadoop.security.AccessControlException(Permission denied: user=hive_quanxian, access=READ_EXECUTE, inode="/user/hive/warehouse/bigdata.db/gd_ga_wa_swryswjl":zhongao:hive:drwx------
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkAccessAcl(FSPermissionChecker.java:426)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(FSPermissionChecker.java:329)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkSubAccess(FSPermissionChecker.java:300)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:241)
at com.xxx.hadoop.adapter.hdfs.plugin.HWAccessControlEnforce.checkPermission(HWAccessControlEnforce.java:69)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:190)
at org.apache.hadoop.hdfs.server.namenode.FSDirectory.checkPermission(FSDirectory.java:1910)
at org.apache.hadoop.hdfs.server.namenode.FSDirectory.checkPermission(FSDirectory.java:1894)
at org.apache.hadoop.hdfs.server.namenode.FSDirStatAndListingOp.getContentSummary(FSDirStatAndListingOp.java:135)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.getContentSummary(FSNamesystem.java:3983)
at org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.getContentSummary(NameNodeRpcServer.java:1342)
at
```

```
org.apache.hadoop.hdfs.protocolPB.ClientNamenodeProtocolServerSideTranslatorPB.getC
ontentSummary(ClientNamenodeProtocolServerSideTranslatorPB.java:925)
at
org.apache.hadoop.hdfs.protocol.proto.ClientNamenodeProtocolProtos$ClientNamenodePr
otocol$2.callBlockingMethod(ClientNamenodeProtocolProtos.java)
at
org.apache.hadoop.ipc.ProtobufRpcEngine$Server$ProtoBufRpcInvoker.call(ProtobufRpcE
ngine.java:616)
at org.apache.hadoop.ipc.RPC$Server.call(RPC.java:973)
at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2260)
at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2256)
at java.security.AccessController.doPrivileged(Native Method)
at javax.security.auth.Subject.doAs(Subject.java:422)
at
org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1781)
at org.apache.hadoop.ipc.Server$Handler.run(Server.java:2254)
)'
```

原因分析

1. 根据堆栈信息，可以看出在检查子目录的权限时失败。

   ```
   org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkSubAccess(FSPer
   missionChecker.java:300)
   ```

2. 检查 HDFS 上表目录下所有文件目录的权限，发现有一个目录权限为 700（只有文件属主能够访问），确认存在异常目录。

解决办法

1. 确认该文件是否为手动异常导入，如不是数据文件或目录，删除该文件或目录。
2. 当无法删除时，建议修改文件或目录权限为 770。

10.39 Load 数据到 Hive 表失败

问题背景与现象

用户在建表成功后，通过 Load 命令往此表导入数据，但导入操作中遇到如下问题：

```
.......
> LOAD DATA INPATH '/user/tester1/hive-data/data.txt' INTO TABLE employees info;
Error: Error while compiling statement: FAILED: SemanticException Unable to load
data to destination table. Error: The file that you are trying to load does not
match the file format of the destination table. (state=42000,code=40000)
..........
```

原因分析

1. 经分析，发现在建表时没有指定存储格式，所以采用了缺省存储格式 RCFile。
2. 在导入数据时，被导入数据格式是 TEXTFILE 格式，最终导致此问题。

解决办法

属于应用侧问题，解决办法有多种。只要保证表所指定存储格式和被导入数据格式是一致的，可以根据实际情况采用合适方法。

- 方法 1：

  可以使用具有 Hive 表操作权限的用户在建表时指定存储格式，例如：

  CREATE TABLE IF NOT EXISTS employees_info(name STRING,age INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE;

  指定被导入数据格式为 TEXTFILE。

- 方法 2：

  被导入数据存储格式不能为 TEXTFILE，而应为 RCFile。

10.40 HiveServer 和 HiveHCat 进程故障

用户问题

客户集群 HiveServer 和 WebHCat 进程状态均为故障。

问题现象

客户 MRS 集群 Master2 节点上的 HiveServer 和 WebHCat 进程状态显示为故障，重启之后仍为故障状态。

原因分析

在 Manager 界面单独启动故障的 hiveserver 进程，登录后台查找 hiveserver.out 日志中对应时间点的报错，报错信息为：error parsing conf mapred-site.xml 和 Premature end of file。然后重启 webhcat 也发现同样报错，原因即为解析 mapred-site.xml 文件错误。

处理步骤

1. 以 root 用户登录 Master2 节点。
2. 执行 **find / -name 'mapred-site.xml'**命令获取 mapred-site.xml 文件所在位置。
   - hiveserver 对应路径为/opt/Bigdata/*集群版本*/1_13_HiveServer/etc/mapred-site.xml，
   - webhcat 对应路径为/opt/Bigdata/*集群版本*/1_13_WebHCat/etc/mapred-site.xml。
3. 确认 mapred-site.xml 文件是否有异常，该案例中该配置文件内容为空导致解析失败。
4. 修复 mapred-site.xml 文件，将 Master1 节点上对应目录下的配置文件用 scp 命令拷贝到 Master2 节点对应目录替换原文件.
5. 执行 **chown omm:wheel mapred-site.xml** 命令更改所属组和用户。
6. 在 Manager 界面重启故障的 HiveServer 和 WebHCat 进程，恢复正常。

10.41 Hive 执行 insert into 语句报错，命令界面报错信息不明

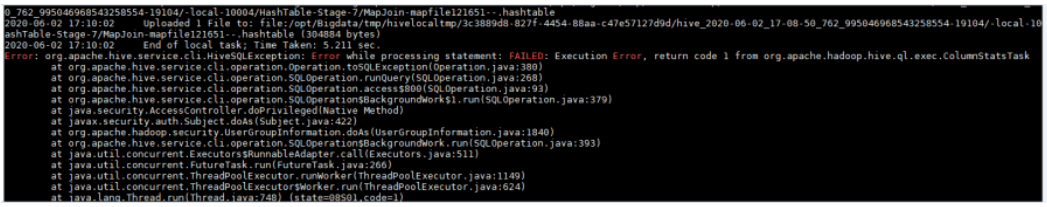
用户问题

客户使用 MRS Hive 执行一条 SQL 报错。

问题现象

客户使用 MRS Hive 执行一条 SQL，有如下报错：

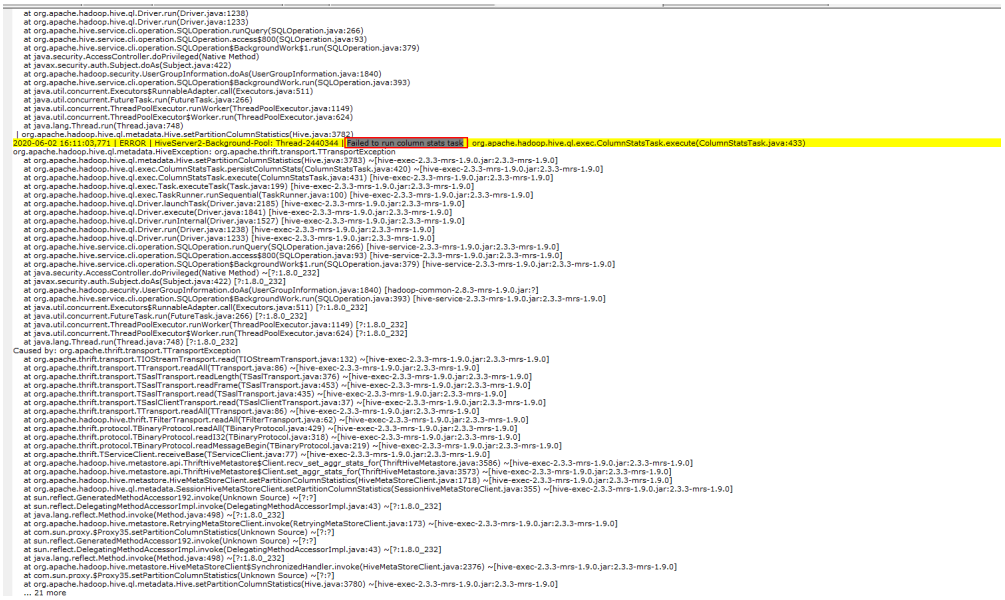
图10-1 使用 MRS Hive 执行 SQL 报错



原因分析

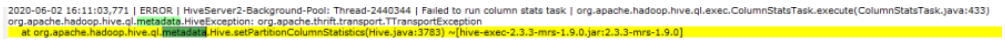
1. 查看 Hiveserver 日志，在对应时间点，有如下的报错信息。

   图10-2 Hiveserver 日志



2. 在如上报错信息中未发现重要信息，但从堆栈中发现 metadata 字样，怀疑报错是和 metastore 有关。

图10-3 堆栈中 metadata 字样



3. 查看 metastore 日志，发现如下报错。

图10-4 metastore 日志

查看如上错误的上下文，确定是本次执行 SQL 的报错，在报错信息里面发现如下内容：

```
Caused by: org.postgresql.util.PSQLException: ERROR: value too long for type
character varying(4000)
```

确认是客户该条 SQL 对表的操作，所有列的字节长度超过 4000 的限制，导致 SQL 执行失败，需要修改该限制。

处理步骤

步骤 1 以 root 用户登录集群任意一个 Master 节点，并执行 **su - omm** 命令切换到 omm 用户。

步骤 2 执行如下命令登录高斯 DB。

gsql -p 20051 -d hivemeta -U *username* -W *password*

步骤 3 执行如下命令修改限制。

alter table PARTITION_PARAMS alter column PARAM_VALUE type varchar(6000);

----结束

10.42 增加 Hive 表字段超时

用户问题

增加 Hive 表字段报错。

问题现象

Hive 对包含 10000+分区的表执行 **ALTER TABLE table_name ADD COLUMNS(column_name string) CASCADE;**，报错如下：

```
Timeout when executing method: alter table with environment context; 600525ms
exceeds 600000ms
```

原因分析

1. MetaStore 客户端连接超时，MRS 默认 MetaStore 客户端和服务端连接的超时时间是 600s，在 Manager 页面调大 hive.metastore.client.socket.timeout 为 3600s。
2. 出现另一个报错：

   ```
   Error: org.apache.hive.service.cli.HiveSQLException: Error while processing
   statement: FAILED: Execution Error, return code 1 from
   org.apache.hadoop.hive.ql.exec.DDLTask. Unable to alter table.
   java.net.SocketTimeoutException: Read timed out
   ```

   Metastore 元数据 JDBC 连接超时，默认 60ms。
3. 调大 javax.jdo.option.ConnectionURL 中 socketTimeout=60000，仍然产生最初的报错：

   ```
   Timeout when executing method: alter_table_with_environment_context;3600556ms
   exceeds 3600000ms
   ```
4. 尝试调大 hive.metastore.batch.retrieve.max、hive.metastore.batch.retrieve.table.partition.max、dbservice.database.max.connections 等参数均未能解决。
5. 怀疑是 GaussDB 的问题，因为增加字段会遍历每个分区执行 getPartitionColumnStatistics 和 alterPartition。
6. 使用 omm 用户执行 **gsql -p 20051 -U omm -W dbserverAdmin@123 -d hivemeta** 登录 hive 元数据库。
7. 执行 **select * from pg_locks;**没有发现锁等待。
8. 执行 **select * from pg_stat_activity;**发现进程执行时间较长。

   ```
   SELECT 'org.apache.hadoop.hive.metastore.model.MPartitionColumnStatistics'AS
   NUCLEUS_TYPE,A0.AVG_COL_LEN,A0."COLUMN_NAME",A0.COLUMN_TYPE,A0.DB_NAME,A0.BIG_D
   ECIMAL_HIGH_VALUE,A0.BIG_DECIMAL_LOW_VALUE,A0.DOUBLE_HIGH_VALUE,A0.DOUBLE_LOW_V
   ALUE,A0.LAST_ANALYZED,A0.LONG_HIGH_VALUE,A0.LONG_LOW_VALUE,A0.MAX_COL_LEN,A0.NU
   M_DISTINCTS,A0.NUM_FALSES,A0.NUM_NULLS,A0.NUM_TRUES,A0.PARTITION_NAME,A0."TABLE
   _NAME",A0.CS_ID,A0.PARTITION_NAMEAS NUCORDER0 FROM PART_COL_STATS A0 WHERE
   A0."TABLE_NAME" = '$1' ANDA0.DB_NAME = '$2' AND A0.PARTITION_NAME = '$3'
   AND((((((A0."COLUMN_NAME" = '$4') OR (A0."COLUMN_NAME" ='$5')) OR
   (A0."COLUMN_NAME" = '$6')) OR (A0."COLUMN_NAME" ='$7')) OR (A0."COLUMN_NAME" =
   '$8')) OR (A0."COLUMN_NAME" ='$9')) ORDER BY NUCORDER0;
   ```

9. 执行 **gs_guc reload -c log_min_duration_statement=100 -D /srv/BigData/dbdata_service/data/**开启 sql 录制，发现 8.执行 select * from pg_sta... 中语句执行时长 700ms，而且因为有 10000+分区，会触发执行 10000+次命令。
10. 在 sql 前加 explain (analyze,verbose,timing,costs,buffers)分析执行计划，发现执行时需要全表扫描。

11. 查看索引，发现不满足最左匹配原则。

```
HIVEMETA=# \d+ PART_COL_STATS
                                    Table "PUBLIC.PART_COL_STATS"
        Column         |          Type          |          Modifiers          | Storage  | Stats target | Description
-----------------------+------------------------+-----------------------------+----------+--------------+-------------
 CS_ID                 | BIGINT                 | not null                    | plain    |              |
 CAT_NAME              | CHARACTER VARYING(256) | default NULL::CHARACTER VARYING | extended |              |
 DB_NAME               | CHARACTER VARYING(128) | default NULL::CHARACTER VARYING | extended |              |
 TABLE_NAME            | CHARACTER VARYING(256) | default NULL::CHARACTER VARYING | extended |              |
 PARTITION_NAME        | CHARACTER VARYING(767) | default NULL::CHARACTER VARYING | extended |              |
 COLUMN_NAME           | CHARACTER VARYING(767) | default NULL::CHARACTER VARYING | extended |              |
 COLUMN_TYPE           | CHARACTER VARYING(128) | default NULL::CHARACTER VARYING | extended |              |
 PART_ID               | BIGINT                 | not null                    | plain    |              |
 LONG_LOW_VALUE        | BIGINT                 |                             | plain    |              |
 LONG_HIGH_VALUE       | BIGINT                 |                             | plain    |              |
 DOUBLE_LOW_VALUE      | DOUBLE PRECISION       |                             | plain    |              |
 DOUBLE_HIGH_VALUE     | DOUBLE PRECISION       |                             | plain    |              |
 BIG_DECIMAL_LOW_VALUE | CHARACTER VARYING(4000) | default NULL::CHARACTER VARYING | extended |              |
 BIG_DECIMAL_HIGH_VALUE | CHARACTER VARYING(4000) | default NULL::CHARACTER VARYING | extended |              |
 NUM_NULLS             | BIGINT                 | not null                    | plain    |              |
 NUM_DISTINCTS         | BIGINT                 |                             | plain    |              |
 BIT_VECTOR            | BYTEA                  |                             | extended |              |
 AVG_COL_LEN           | DOUBLE PRECISION       |                             | plain    |              |
 MAX_COL_LEN           | BIGINT                 |                             | plain    |              |
 NUM_TRUES             | BIGINT                 |                             | plain    |              |
 NUM_FALSES            | BIGINT                 |                             | plain    |              |
 LAST_ANALYZED         | BIGINT                 | not null                    | plain    |              |
Indexes:
    "PART_COL_STATS_pkey" PRIMARY KEY, BTREE (CS_ID)
    "PART_COL_STATS_N49" BTREE (PART_ID)
    "PCS_STATS_IDX" BTREE (CAT_NAME, DB_NAME, TABLE_NAME, COLUMN_NAME, PARTITION_NAME)
Foreign-key constraints:
    "PART_COL_STATS_fkey" FOREIGN KEY (PART_ID) REFERENCES PARTITIONS(PART_ID) DEFERRABLE
Has OIDs: no
```

处理步骤

1. 重建索引。

```
su - omm
gsql -p 20051 -U omm -W dbserverAdmin@123 -d hivemeta
DROP INDEX PCS_STATS_IDX;
```

```
CREATE INDEX PCS_STATS_IDX ON PART_COL_STATS(DB_NAME, TABLE_NAME, COLUMN_NAME,
PARTITION_NAME);
CREATE INDEX SDS_N50 ON SDS(CD_ID);
```

2. 重新查看执行计划，发现语句已经可以索引查询，且 5ms 执行完成（原来是 700ms）。重新执行 hive 表字段增加，已经可以添加成功。

10.43 Hive 服务重启失败

用户问题

修改 Hive 服务配置后，保存配置失败，Manager 页面 Hive 服务的配置状态为配置失败。

问题现象

用户 A 在 MRS 节点后台上打开了 Hive 相关配置文件且未关闭，此时用户 B 在 MRS Manager 页面的“服务管理”中修改 Hive 配置项，保存配置并重启 Hive 服务，此时保存配置失败，并且 Hive 服务启动失败。

原因分析

由于用户 B 在 MRS Manager 页面修改配置时，配置文件被用户 A 在 MRS 节点后台打开，导致该配置文件不能被替换，最终导致 Hive 服务启动失败。

处理步骤

步骤 1 用户需要首先手动关闭集群节点后台打开的 Hive 配置文件。

步骤 2 在 MRS Manager 页面重新修改 Hive 的配置并保存配置。

步骤 3 重启 Hive 服务。

----结束

10.44 hive 执行删除表失败

用户问题

hive 表删除失败

问题现象

hive 创建的二级分区表有两万多个分区，导致用户在执行 **truncate table ${TableName},drop table ${TableName}**时失败。

原因分析

删除文件操作是单线程串行执行的，hive 分区数过多导致在元数据数据库会保存大量元数据信息，在执行删表语句时删除元数据就要用很长时间，最终在超时时间内删除不完，就会导致操作失败。

说明

超时时间可通过登录 FusionInsight Manager，选择“集群 > 服务 > Hive > 配置 >全部配置 > MetaStore（角色） > 服务初始化”查看，“hive.metastore.client.socket.timeout”对应的值即为超时时间时长，在“描述”列可查看默认值。

处理步骤

步骤 1 如果是内部表可以先通过 **alter table ${TableName} set TBLPROPERTIES('EXTERNAL'='true')**来将内部表转成外部表，这样 hive 删除的时候只删除元数据省去了删除 hdfs 数据的时间。

步骤 2 如果要用相同的表名可以先将表结构用 **show create table ${TableName}**来导出表结构，再用 **ALTER TABLE ${TableName} RENAME TO ${new_table_name};**来将表重命名。这样就可以新建一个和原来一样表。

步骤 3 执行 **hdfs dfs －rm －r －f ${hdfs_path}**在 hdfs 上删除表数据。

步骤 4 在 hive 中用 **alter table ${Table_Name} drop partition (${PartitionName}<’XXXX’, ${PartitionName}>’XXXX’);**删除分区(具体删除条件可灵活处理),减少文件数。

步骤 5 删除分区少于一千个后，直接用 **drop table ${TableName}**删掉表即可。

----结束

建议与总结

hive 分区虽然可以提高查询效率,但要避免分区不合理导致出现大量小文件的问题,要提前规划好分区策略.

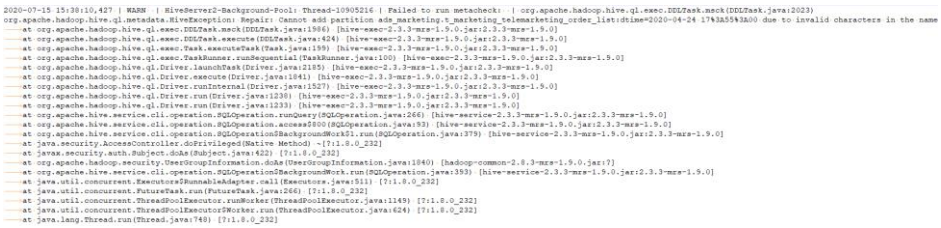
10.45 Hive 执行 msck repair table table_name 报错

现象描述

Hive 执行 msck repair table table_name 报错：FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.ql.exec.DDLTask (state=08S01,code=1)。

可能原因

查看 HiveServer 日志/var/log/Bigdata/hive/hiveserver/hive.log，发现目录名不符合分区格式。



处理步骤

- 方法一：删除错误的文件或目录。
- 方法二：执行 set hive.msck.path.validation=skip，跳过无效的目录。

10.46 在 Hive 中 drop 表后，如何完全释放磁盘空间

用户问题

在 Hive 命令行执行 **drop** 表的操作后，通过命令 **hdfs dfsadmin -report** 查看磁盘空间，发现表没有删除。

原因分析

在 Hive 命令行执行 **drop** 表只删除了外部表的表结构，并没有删除该表存储在 HDFS 上的表数据。

处理步骤

步骤 1 使用 **root** 用户登录安装客户端的节点，并认证用户。

cd *客户端安装目录*

source bigdata_env

kinit *组件业务用户*（未开启 Kerberos 认证的集群跳过此操作）

步骤 2 执行以下命令删除存储在 HDFS 上的表。

hadoop fs -rm hdfs://hacluster/*表所在的具体路径*

----结束

10.47 客户端执行 SQL 报错连接超时

现象描述

客户端执行 SQL 失败，报错：Timed out waiting for a free available connection。

可能原因

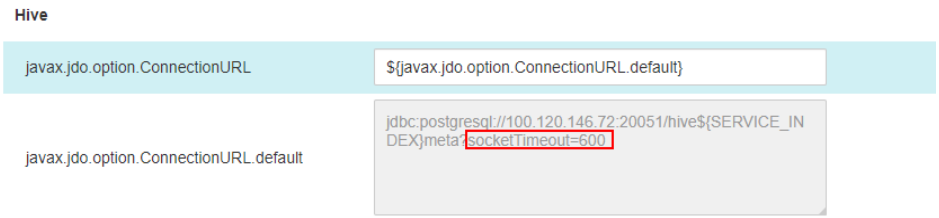
DBService 连接较多，获取连接超时。

操作步骤

步骤 1 客户端是否使用 Spark-SQL 客户端执行 SQL。

- 是，检查连接的 URL 中超时参数，将其修改为 600，执行步骤 7。
- 否，执行步骤 2。

步骤 2 登录 Manager 页面，选择“集群 > 服务 > Hive > 配置 > 全部配置”，搜索“javax.jdo.option.ConnectionURL”，检查超时参数是否小于 600。



说明

Hive、HiveServer、MetaStore、WebHCat 中均有该参数，请确保它们的参数值一致。

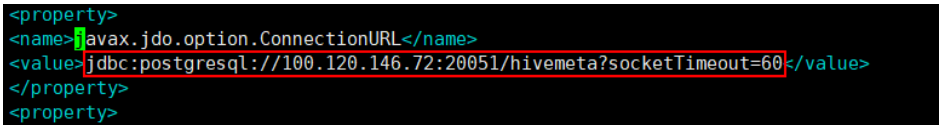
- 是，执行步骤 3。
- 否，执行步骤 7。

步骤 3 检查参数“javax.jdo.option.ConnectionURL”的值是否为“${javax.jdo.option.ConnectionURL.default}”。

- 是，执行步骤 4。
- 否，修改 URL 中超时参数为 600，单击“保存”，执行步骤 7

步骤 4 单击“实例”，选择任意 HiveServer 实例，并使用 **root** 用户登录实例节点。

步骤 5 打开配置文件“${BIGDATA_HOME}/FusionInsight_Current/*HiveServer/etc/hivemetastore-site.xml”，查找配置项“javax.jdo.option.ConnectionURL”，复制配置项值。



步骤 6 登录 Manager 页面，选择“集群 > 服务 > Hive > 配置 > 全部配置”，搜索“javax.jdo.option.ConnectionURL”，修改配置为步骤 5 中复制的 URL，并修改超时参数为 600，单击“保存”。

说明

Hive、HiveServer、MetaStore、WebHCat 中均有该参数，请确保它们的参数值一致。

步骤 7 选择“集群 > 服务 > Hive > 配置 > 全部配置”，搜索“maxConnectionsPerPartition”，检查是否小于 100。

- 是，修改参数为 100，单击“保存”，执行步骤 8。
- 否，执行步骤 8。

步骤 8 若以上步骤有修改参数，选择“集群 > 服务 > Hive > 概览”，选择“更多 > 滚动重启服务”，未修改则无需执行此步骤。

----结束

10.48 WebHCat 健康状态异常导致启动失败

用户问题

WebHCat 实例启动失败。

问题现象

在 Manager 页面上查看到 WebHCat 实例的健康状态为“故障”，并上报“ALM-12007 进程故障”告警，该告警的服务名称为“Hive”，实例名称为“WebHCat”。且重启 Hive 服务报错。

查看 WebCat 实例的日志“/var/log/Bigdata/hive/webhcat/webhcat.log”报错“Service not found in Kerberos database”和“Address already in use”。

处理步骤

步骤 1 依次登录 WebHCat 实例所在节点检查“/etc/hosts”文件中的 IP 及主机名称映射关系是否正确。且“/etc/hostname”和“/etc/HOSTNAME”文件的 WebHCat 配置需与“/etc/hosts”保持一致，若不一致则需手动修改。

说明

WebHCat 实例的 IP 地址及主机名称映射关系可登录 FusionInsight Manager 界面，选择“集群 > 服务 > Hive > 实例”查看。

步骤 2 登录 WebHCat 实例所在节的任一节点，执行以下命令切换到 **omm** 用户。

su - omm

步骤 3 执行以下命令查看是否存在 WebHCat 进程。

ps -ef|grep webhcat|grep -v grep

若存在，则需执行以下命令结束 WebHCat 进程：

kill -9 ${webhcat_pid}

步骤 4 登录 FusionInsight Manager，选择“集群 > 服务 > Hive > 实例”，勾选所有 WebHCat 实例，选择“更多 > 重启实例”，等待 WebHCat 重启成功即可。

----结束

10.49 mapred-default.xml 文件解析异常导致 WebHCat 启动失败

用户问题

MRS 的 Hive 服务故障，重新启动后，Master2 节点上的 HiveServer 和 WebHCat 进程启动失败，Master1 节点进程正常。

原因分析

登录 Master2 节点，查看“/var/log/Bigdata/hive/hiveserver/hive.log”日志，发现 HiveServer 一直加载“/opt/Bigdata/*/*_HiveServer/etc/hive-site.xml”；查看 HiveServer 退出时的“/var/log/Bigdata/hive/hiveserver/hiveserver.out”日志，发现解析“mapred-default.xml”文件异常。

处理步骤

步骤 1 登录 Master2 节点，使用以下命令查找“mapred-default.xml”所在路径：

find /opt/ -name 'mapred-default.xml'

查询到该配置文件在“/opt/Bigdata/*/*_WebHCat/etc/”目录下面，且该文件内容为空。

步骤 2 登录到 Master1 节点，将“/opt/Bigdata/*/*_WebHCat/etc/mapred-default.xml”文件拷贝到 Master2 节点，并修改文件的属组为“omm:wheel”。

步骤 3 登录 Manager，重启异常的 HiveServer 和 WebHCat 实例。

----结束

11 使用 Hue

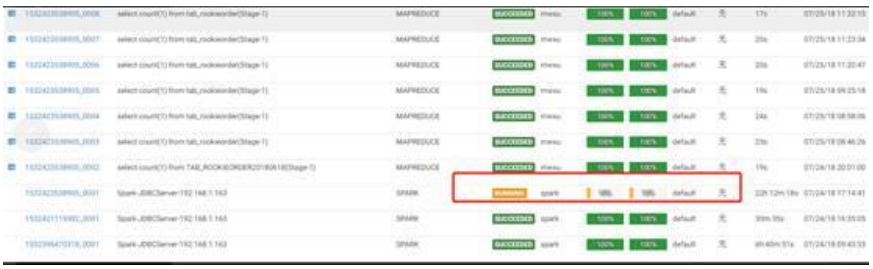
11.1 Hue 上有 job 在运行

用户问题

客户查到 Hue 上有 job 在运行。

问题现象

客户的 MRS 装好后， Hue 上查到有 Job 在运行 ，并且目前在运行的 job 并不是客户操作的。



原因分析

此 Job 为 Spark 启动之后，系统自身连接 jdbc 的一个任务，是常驻的。

处理步骤

非问题，无需处理。

11.2 使用 IE 浏览器在 Hue 中执行 HQL 失败

问题背景与现象

使用 IE 浏览器在 Hue 中访问 Hive Editor 并执行所有 HQL 失败，界面提示“There was an error with your query.”。

原因分析

IE 浏览器存在功能问题，不支持在 307 重定向中处理含有 form data 的 AJAX POST 请求，建议更换兼容的浏览器。

解决办法

使用 Google Chrome 浏览器 21 及以上版本。

11.3 Hue WebUI 访问失败

用户问题

访问 Hue WebUI 跳转到错误的页面。

问题现象

查看 hue web ui 报错如下：

```
503 Service Unavailable
The server is temporarily unable to service your requster due to maintenance
downtime or capacity problems.Please try again later.
```

原因分析

- hue 配置过期。
- 单 Master 节点集群中，Hue 服务需要手动修改配置。

处理步骤

步骤 1 登录 Master 节点。

步骤 2 执行 **hostname -i** 获取本机 IP。

步骤 3 执行如下命令获取“HUE_FLOAT_IP”的地址：

grep "HUE_FLOAT_IP" ${BIGDATA_HOME}/*MRS*_Current/1_*/etc*/ENV_VARS,其中 *MRS* 以实际文件名为准。

步骤 4 比较本机 IP 和“HUE_FLOAT_IP”的值是否相同，若不相同，请修改“HUE_FLOAT_IP”的值为本机 IP。

步骤 5 重启 Hue 服务。

----结束

11.4 Hue 界面无法加载 HBase 表

用户问题

用户在 Hue 界面将 hive 数据导入 hbase 后，报检测不到 hbase 表的错误。

问题现象

Kerberos 集群中，IAM 子账户权限不足导致无法加载 hbase 表。

原因分析

IAM 子账户权限不足。

处理步骤

MRS Manager 界面操作：

步骤 1 登录 MRS Manager。

步骤 2 选择“系统管理 > 用户管理”。

步骤 3 在使用的用户所在行的单击“修改”。

步骤 4 为用户添加 supergroup 组。

步骤 5 单击“确定”完成修改操作。

----结束

FusionInsight Manager 界面操作：

步骤 1 登录 FusionInsight Manager。

步骤 2 选择“系统 > 权限 > 用户”。

步骤 3 在使用的用户所在行单击“修改”。

步骤 4 为用户添加 supergroup 组。

步骤 5 单击“确定”完成修改操作。

----结束

建议与总结

如果是开启 Kerberos 认证的集群，页面出现 No data available 优先排查权限问题。

12 使用 Impala

12.1 用户连接 impala-shell 失败

用户问题

用户连接 impala-shell 失败。

问题现象

用户在“组件管理”页面修改任意组件的配置并重启服务后，连接 impala-shell，会出现连接失败，报错 no such file/directory。

```
[root@node-masterlemdj etc]# pwd
/opt/Bigdata/MRS_2.1.0/1_7_KuduMaster/etc
[root@node-masterlemdj etc]# impala-shell -i 192.168.0.73
shell-init: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory
chdir: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory
Traceback (most recent call last):
  File "/opt/client/Impala/impala/shell/impala_shell.py", line 38, in <module>
    from impala_client import (ImpalaClient, DisconnectedException, QueryStateException,
  File "/opt/client/Impala/impala/shell/lib/impala_client.py", line 20, in <module>
    import sasl
  File "build/bdist.linux-x86_64/egg/sasl/__init__.py", line 1, in <module>

  File "build/bdist.linux-x86_64/egg/sasl/saslwrapper.py", line 7, in <module>
  File "build/bdist.linux-x86_64/egg/_saslwrapper.py", line 7, in <module>
  File "build/bdist.linux-x86_64/egg/_saslwrapper.py", line 3, in __bootstrap__
  File "/usr/lib/python2.7/site-packages/setuptools-0.6c11-py2.7.egg/pkg_resources.py", line 2594, in <module>
    for comparator, version in req.specs:
  File "/usr/lib/python2.7/site-packages/setuptools-0.6c11-py2.7.egg/pkg_resources.py", line 425, in __init__

  File "/usr/lib/python2.7/site-packages/setuptools-0.6c11-py2.7.egg/pkg_resources.py", line 440, in add_entry
    `req`.  But, if there is an active distribution for the project and it
  File "/usr/lib/python2.7/site-packages/setuptools-0.6c11-py2.7.egg/pkg_resources.py", line 1688, in find_on_path
    return ()
  File "/usr/lib/python2.7/site-packages/setuptools-0.6c11-py2.7.egg/pkg_resources.py", line 1835, in _normalize_cached

  File "/usr/lib/python2.7/site-packages/setuptools-0.6c11-py2.7.egg/pkg_resources.py", line 1829, in normalize_path
    register_namespace_handler(object,null_ns_handler)
  File "/usr/lib64/python2.7/posixpath.py", line 368, in realpath
    return abspath(path)
  File "/usr/lib64/python2.7/posixpath.py", line 356, in abspath
    cwd = os.getcwd()
OSError: [Errno 2] No such file or directory
```

原因分析

修改服务配置并重启服务后，部分服务的目录结构会删除并重新创建，如服务的 etc 目录等。如果重启服务前所在的目录为 etc 或者其子目录，由于重启后目录重建，仍在原来目录执行 impala-shell 时会产生某些系统变量或者参数无法找到的情况，所以连接 impala-shell 连接失败。

处理步骤

任意切换到存在的目录，重新连接 impala-shell 即可。

12.2 创建 Kudu 表报错

用户问题

创建 Kudu 表报错。

问题现象

新建了集群，在创建表时，报错“[Cloudera]ImpalaJDBCDriver ERROR processing query/statement. Error Code: 0, SQL state: TStatus(statusCode:ERROR_STATUS, sqlState:HY000, errorMessage:AnalysisException: Table property 'kudu.master_addresses' is required when the impalad startup flag -kudu_master_hosts is not used.”

原因分析

客户未在 impala sql 中指定 kudu.master_addresses 地址导致报错：Table property 'kudu.master_addresses' is required when the impalad startup flag -kudu_master_hosts is not used.

处理步骤

在创建 Kudu 表时指定“kudu.master_addresses”地址。

12.3 Impala 客户端登录失败

用户问题

运行 Impala client 会报类似如下错误信息：

```
[root@node-master1avIy ~]# impala-shell -i 192.168.128.49:21000
  File "/opt/client/Impala/impala/shell/impala_shell.py", line 1675
    except Exception, e:
                    ^
SyntaxError: invalid syntax
[root@node-master1avIy ~]#
```

原因分析

由于最新的 MRS 集群使用的是 Euler2.9 及以上版本的操作系统，系统自带只 python3 版本，而 Impala client 是基于 python2 实现的，和 python3 部分语法不兼容，运行 Impala client 会报错误信息，所以需要手动安装 python2 以解决 Impala client 运行问题。

处理步骤

步骤 1 使用 root 用户登录 Impala 所在节点，执行如下命令，确认当前系统上安装的 python 版本：

python --version

```
[root@node-master2JgOY ~]# python --version
Python 3.7.4
```

步骤 2 执行命令 **yum install make**，查看 yum 是否可用。

- 如果 yum install 报如下错误，说明 yum 设置有问题，执行步骤 3。

```
[root@node-master2JgOY ~]# yum install make
Error: There are no enabled repositories in "/etc/yum.repos.d", "/etc/yum/repos.d", "/etc/distro.repos.d".
```

- 如果没有报错，执行步骤 4。

步骤 3 执行命令 **cat /etc/yum.repos.d/EulerOS-base.repo**，查看 yum 源和系统版本信息不匹配是否匹配，如果不匹配则修改，如下所示：

修改前：

```
[root@node-master1avIy ~]# cat /etc/yum.repos.d/EulerOS-base.repo
[base]
name=EulerOS-2.0SP2 base
baseurl=http://mirrors.[redacted].com/euler/ict/site-euleros/euleros/repo/yum/2.2/os/x86_64/
enabled=1
gpgcheck=1
gpgkey=http://mirrors.[redacted].com/euler/ict/site-euleros/euleros/repo/yum/2.2/os/RPM-GPG-KEY-EulerOS
[root@node-master1avIy ~]# uname -a
Linux node-master1avIy.mrs-mq7v.com 4.18.0-147.5.1.6.h541.eulerosv2r9.x86_64 #1 SMP Wed Aug 4 02:30:13 UTC
 x86_64 GNU/Linux
```

修改后：

```
[base]
name=EulerOS-2.0SP9 base
baseurl=http://mirrors.[redacted].com/euler/ict/site-euleros/euleros/repo/yum/2.9/os/x86_64/
enabled=1
gpgcheck=1
gpgkey=http://mirrors.[redacted].com/euler/ict/site-euleros/euleros/repo/yum/2.9/os/RPM-GPG-KEY-EulerOS
```

步骤 4 执行如下命令，查看 yum 源上 python2 开头的软件。

yum list python2*

```
[root@node-master2JgOY ~]# yum list python2*
Last metadata expiration check: 0:02:36 ago on Thu 16 Dec 2021 10:05:52 AM CST.
Available Packages
python2.x86_64                                          2.7.16-16.eulerosv2r9
python2-debug.x86_64                                    2.7.16-16.eulerosv2r9
python2-devel.x86_64                                    2.7.16-16.eulerosv2r9
python2-help.noarch                                     2.7.16-16.eulerosv2r9
python2-pip.noarch                                      18.0-13.h2.eulerosv2r9
python2-setuptools.noarch                               40.4.3-4.h1.eulerosv2r9
python2-tkinter.x86_64                                  2.7.16-16.eulerosv2r9
python2-tools.x86_64                                    2.7.16-16.eulerosv2r9
```

步骤 5 执行如下命令，安装 python2。

yum install python2

```
[root@node-master2JgOY ~]# yum install python2
Last metadata expiration check: 0:00:48 ago on Thu 16 Dec 2021 10:05:52 AM CST.
Error:
 Problem: problem with installed package python3-unversioned-command-3.7.4-7.h29.eulerosv2r9.x86_64
  - package python3-unversioned-command-3.7.4-7.h29.eulerosv2r9.x86_64 conflicts with python2 provided by python2-2.7.16-16.e
ulerosv2r9.x86_64
  - package python3-unversioned-command-3.7.4-7.h11.eulerosv2r9.x86_64 conflicts with python2 provided by python2-2.7.16-16.e
ulerosv2r9.x86_64
  - package python3-unversioned-command-3.7.4-7.h13.eulerosv2r9.x86_64 conflicts with python2 provided by python2-2.7.16-16.e
ulerosv2r9.x86_64
  - package python3-unversioned-command-3.7.4-7.h15.eulerosv2r9.x86_64 conflicts with python2 provided by python2-2.7.16-16.e
ulerosv2r9.x86_64
  - package python3-unversioned-command-3.7.4-7.h18.eulerosv2r9.x86_64 conflicts with python2 provided by python2-2.7.16-16.e
ulerosv2r9.x86_64
  - package python3-unversioned-command-3.7.4-7.h33.eulerosv2r9.x86_64 conflicts with python2 provided by python2-2.7.16-16.e
ulerosv2r9.x86_64
  - package python3-unversioned-command-3.7.4-7.h38.eulerosv2r9.x86_64 conflicts with python2 provided by python2-2.7.16-16.e
ulerosv2r9.x86_64
  - conflicting requests
(try to add '--allowerasing' to command line to replace conflicting packages or '--skip-broken' to skip uninstallable package
s or '--nobest' to use not only best candidate packages)
```

因为当前系统上已安装 python3，所有直接安装 python2 会有上面的冲突提示。

可以选择--allowerasing 或--skip-broken 安装，例如：

yum install python2 --skip-broken

```
[root@node-master2JgOY ~]# yum install python2 --skip-broken
Last metadata expiration check: 0:34:08 ago on Thu 16 Dec 2021 10:05:52 AM CST.
Dependencies resolved.
==============================================================================================================================
 Package                            Architecture        Version                                     Repository          Size
==============================================================================================================================
Installing:
 python2                            x86_64              2.7.16-16.eulerosv2r9                       base               6.4 M
Installing dependencies:
 libXft                             x86_64              2.3.2-13.eulerosv2r9                        base                41 k
```

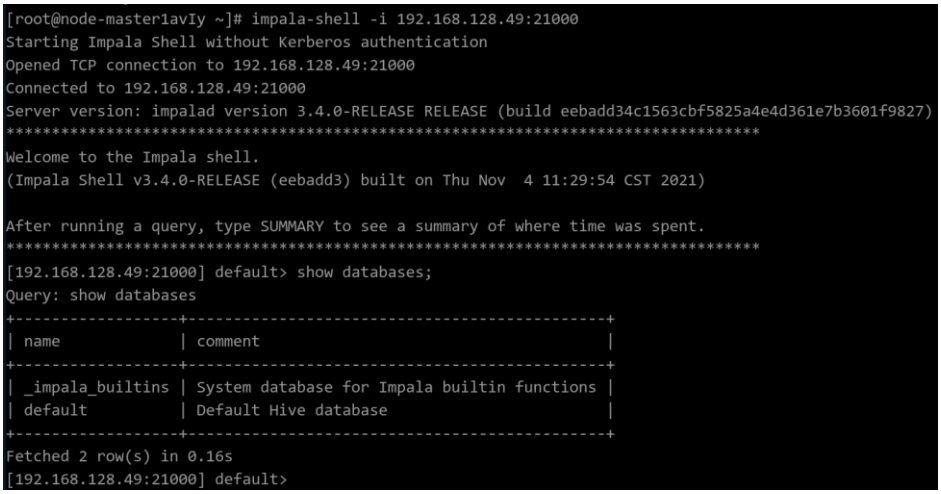
安装完成后，会自动将 python 版本修改为 python2，如下所示：

```
Installed:
  libXft-2.3.2-13.eulerosv2r9.x86_64                                 libXrender-0.9.10-10.eulerosv2r9.x86_64
  python2-2.7.16-16.eulerosv2r9.x86_64                               python2-debug-2.7.16-16.eulerosv2r9.x86_64
  python2-devel-2.7.16-16.eulerosv2r9.x86_64                         python2-help-2.7.16-16.eulerosv2r9.noarch
  python2-setuptools-40.4.3-4.h1.eulerosv2r9.noarch                  python2-tkinter-2.7.16-16.eulerosv2r9.x86_64
  python2-tools-2.7.16-16.eulerosv2r9.x86_64                         python3-rpm-generators-9-1.eulerosv2r9.noarch
  tk-1:8.6.8-5.eulerosv2r9.x86_64

Complete!
[root@node-master2JgOY ~]# python --version
Python 2.7.16
```

如果 python2 安装成功，但是显示的 python 版本不对，需要执行以下命令手动给“/usr/bin/python2”创建软链接“/usr/bin/python”。

ln -s /usr/bin/python2 /usr/bin/python

步骤 6 验证 Impala client 是否可用。


[root@node-master1avIy ~]# impala-shell -i 192.168.128.49:21000
Starting Impala Shell without Kerberos authentication
Opened TCP connection to 192.168.128.49:21000
Connected to 192.168.128.49:21000
Server version: impalad version 3.4.0-RELEASE RELEASE (build eebadd34c1563cbf5825a4e4d361e7b3601f9827)
Welcome to the Impala shell.
(Impala Shell v3.4.0-RELEASE (eebadd3) built on Thu Nov 4 11:29:54 CST 2021)
After running a query, type SUMMARY to see a summary of where time was spent.
[192.168.128.49:21000] default> show databases;
Query: show databases
name	comment
_impala_builtins	System database for Impala builtin functions
default	Default Hive database
Fetched 2 row(s) in 0.16s
[192.168.128.49:21000] default>

----结束

13 使用 Kafka

13.1 运行 Kafka 获取 topic 报错

用户问题

客户运行 Kafka 获取 topic 报错。

问题现象

运行 Kafka 获取 topic 时报错，报错内容如下：

```
ERROR org.apache.kafka.common.errors.InvalidReplicationFactorException: Replication
factor: 2 larger than available brokers: 0.
```

原因分析

由特殊字符导致获取 zookeeper 地址的变量错误。

处理步骤

步骤 1 登录任意一个 Master 节点。

步骤 2 执行 **cat** *客户端安装目录*/**Kafka/kafka/config/server.properties |grep '^zookeeper.connect =**'命令，查看 zookeeper 地址的变量。

步骤 3 重新运行 Kafka 获取 topic，其中从步骤 2 中获取的变量不要添加任何字符。

----结束

13.2 Flume 可以正常连接 Kafka，但是发送消息失败。

问题现象

使用 MRS 版本安装集群，主要安装 ZooKeeper、Flume、Kafka。

在使用 Flume 向 Kafka 发送数据功能时，发现 Flume 发送数据到 Kafka 失败。

可能原因

1. Kafka 服务异常。
2. Flume 连接 Kafka 地址错误，导致发送失败。
3. Flume 发送超过 Kafka 大小限制的消息，导致发送失败。

原因分析

Flume 发送数据到 Kafka 失败，可能原因是 Flume 侧问题或者 Kafka 侧问题。

1. Manager 界面查看当前 Kafka 状态及监控指标。
   - MRS Manager 界面操作：登录 MRS Manager，选择 “服务管理 > Kafka ”，查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
   - FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka，

     查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
2. 查看 Flume 日志，发现打印 MessageSizeTooLargeException 异常信息，如下所示：

   ```
   2016-02-26 14:55:19,126 | WARN  | [SinkRunner-PollingRunner-
   DefaultSinkProcessor] |  Produce request with correlation id 349829 failed due
   to [LOG,7]: kafka.common.MessageSizeTooLargeException  |
   kafka.utils.Logging$class.warn(Logging.scala:83)
   ```

   通过异常信息，发现当前 Flume 向 Kafka 写入的数据超过了 Kafka 服务端定义的消息的最大值。
3. 通过 Manager 查看 Kafka 服务端定义的消息的最大值。
   - MRS Manager 界面操作入口：登录 MRS Manager，依次选择 “服务管理 > Kafka> 配置”。
   - FusionInsight Manager 界面操作入口：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka > 配置"。

   进入 Kafka 配置页面，参数类别选择全部配置，显示所有 Kafka 相关配置，在“搜索”中输入 message.max.bytes 进行检索。

   MRS 中 Kafka 服务端默认可以接收的消息最大为 1000012 bytes =977KB。

解决办法

与用户确认，当前 Flume 发送数据确实存在超过 1M 的消息。因此，为了确保当前这些消息能够写入 Kafka，需要调整 Kafka 服务端相关参数。

步骤 1 修改 message.max.bytes，使得 message.max.bytes 的值大于当前业务中消息最大值，使得 Kafka 服务端可以接收全部消息。

步骤 2 修改 replica.fetch.max.bytes，使得 **replica.fetch.max.bytes >= message.max.bytes**，使得不同 Broker 上的 Partition 的 Replica 可以同步到全部消息。

- MRS Manager 界面操作入口：登录 MRS Manager，依次选择 “服务管理 > Kafka> 配置”。

- FusionInsight Manager 界面操作入口：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka > 配置"。

进入 Kafka 配置页面，参数类别选择全部配置， 显示所有 Kafka 相关配置，在“搜索”中输入 replica.fetch.max.bytes 进行检索。

步骤 3 单击“保存”，并重启 Kafka 服务，使得 Kafka 相关配置生效。

步骤 4 修改消费者业务应用中 fetch.message.max.bytes，使得 fetch.message.max.bytes >= message.max.bytes，确保消费者可以消费到全部消息。

----结束

13.3 Producer 发送数据失败，抛出 NullPointerException

问题现象

使用 MRS 安装集群，主要安装 ZooKeeper、Kafka。

在使用 Producer 向 Kafka 发送数据功能时，发现客户端抛出 NullPointerException。

可能原因

1. Kafka 服务异常。
2. 客户端 Producer 侧配置 jaas 和 keytab 文件错误。

原因分析

Producer 发送数据到 Kafka 失败，可能原因客户端 Producer 侧问题或者 Kafka 侧问题。

1. 通过 Manager 页面查看 kafka 服务状态及监控指标。
   - MRS Manager 界面操作：登录 MRS Manager，依次选择 "服务管理 > Kafka ，查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
   - FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka，

     查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
2. 查看 Producer 客户端日志，发现打印 NullPointerException 异常信息，如图 13-1 所示。

图13-1 Producer 客户端日志

```
[2016-12-06 02:04:05,906]-[schedule-C50D0717-4643-4D4E-9D6E-B940E4BD7159]-[kafka-producer-network-thread |
SZX1000161910-10.21.219.222-bigdata-producer-5]-[1005]-[org.apache.kafka.clients.producer.internals.Sender.run
thread:
java.lang.NullPointerException
    at org.apache.kafka.common.network.Selector.pollSelectionKeys(Selector.java:302)
    at org.apache.kafka.common.network.Selector.poll(Selector.java:283)
    at org.apache.kafka.clients.NetworkClient.poll(NetworkClient.java:260)
    at org.apache.kafka.clients.producer.internals.Sender.run(Sender.java:229)
    at org.apache.kafka.clients.producer.internals.Sender.run(Sender.java:134)
    at java.lang.Thread.run(Thread.java:745)
[2016-12-06 02:04:05,921]-[schedule-C50D0717-4643-4D4E-9D6E-B940E4BD7159]-[kafka-producer-network-thread |
SZX1000161910-10.21.219.222-bigdata-producer-3]-[1005]-[org.apache.kafka.clients.producer.internals.Sender.run
thread:
java.lang.NullPointerException
    at org.apache.kafka.common.network.Selector.pollSelectionKeys(Selector.java:302)
    at org.apache.kafka.common.network.Selector.poll(Selector.java:283)
    at org.apache.kafka.clients.NetworkClient.poll(NetworkClient.java:260)
    at org.apache.kafka.clients.producer.internals.Sender.run(Sender.java:229)
    at org.apache.kafka.clients.producer.internals.Sender.run(Sender.java:134)
    at java.lang.Thread.run(Thread.java:745)
```

或者日志中只有异常信息没有堆栈信息（只有 NullPointerException 无堆栈信息，出现这个问题是 jdk 的自我保护，相同堆栈打印太多，就会触发这个保护开关，后续不再打印堆栈），如图 13-2 所示。

图13-2 异常信息

```
[2016-11-23 04:06:53,973] [kafka-producer-network-thread | producer-1] [ERROR] [org.apache.kafka.clients.producer.internals.Sender] (run:130)- Uncaught error in kafka producer I/O thread:
java.lang.NullPointerException
[2016-11-23 04:06:53,973] [kafka-producer-network-thread | producer-1] [ERROR] [org.apache.kafka.clients.producer.internals.Sender] (run:130)- Uncaught error in kafka producer I/O thread:
java.lang.NullPointerException
[2016-11-23 04:06:53,973] [kafka-producer-network-thread | producer-1] [ERROR] [org.apache.kafka.clients.producer.internals.Sender] (run:130)- Uncaught error in kafka producer I/O thread:
java.lang.NullPointerException
[2016-11-23 04:06:53,973] [kafka-producer-network-thread | producer-1] [ERROR] [org.apache.kafka.clients.producer.internals.Sender] (run:130)- Uncaught error in kafka producer I/O thread:
java.lang.NullPointerException
```

3. 查看 Producer 客户端日志，发现打印 Failed to configure SaslClientAuthenticator 异常信息，如图 13-3 所示。

图13-3 异常日志信息

```
Caused by: org.apache.kafka.common.KafkaException: Failed to configure SaslClientAuthenticator
at org.apache.kafka.common.security.authenticator.SaslClientAuthenticator.configure(SaslClientAuthenticator.java:96)
at org.apache.kafka.common.network.SaslChannelBuilder.buildChannel(SaslChannelBuilder.java:89)
... 9 more
Caused by: org.apache.kafka.common.KafkaException: Failed to create SaslClient
at org.apache.kafka.common.security.authenticator.SaslClientAuthenticator.createSaslClient(SaslClientAuthenticator.java:112)
at org.apache.kafka.common.security.authenticator.SaslClientAuthenticator.configure(SaslClientAuthenticator.java:94)
... 10 more
Caused by: javax.security.sasl.SaslException: PLAIN: authorization ID and password must be specified
at com.sun.security.sasl.PlainClient.<init>(PlainClient.java:58)
at com.sun.security.sasl.ClientFactoryImpl.createSaslClient(ClientFactoryImpl.java:97)
at javax.security.sasl.Sasl.createSaslClient(Sasl.java:384)
at com.ibm.messagehub.login.MessageHubSaslClientFactory.createSaslClient(MessageHubSaslClientFactory.java:77)
at javax.security.sasl.Sasl.createSaslClient(Sasl.java:384)
at org.apache.kafka.common.security.authenticator.SaslClientAuthenticator$1.run(SaslClientAuthenticator.java:107)
at org.apache.kafka.common.security.authenticator.SaslClientAuthenticator$1.run(SaslClientAuthenticator.java:102)
at java.security.AccessController.doPrivileged(Native Method)
at javax.security.auth.Subject.doAs(Subject.java:422)
at org.apache.kafka.common.security.authenticator.SaslClientAuthenticator.createSaslClient(SaslClientAuthenticator.java:102)
... 11 more
```

4. 认证失败导致创建 KafkaChannel 失败，导致通过 channel(key)方法获取的 KafkaChannel 为空，以至于疯狂打印 NullPointerException，上述日志可以发现，认证失败的原因是用户密码不正确，密码不正确的原因可能是用户名不匹配导致。
5. 检查 Jaas 文件和 Keytab 文件，发现 Jaas 文件中配置使用的 pricipal 为 stream。

图13-4 检查 Jaas 文件

```
KafkaClient {
com.sun.security.auth.module.Krb5LoginModule required
debug=false
keyTab="/opt/client/user.keytab"
useTicketCache=false
storeKey=true
principal="stream@HADOOP.COM"
useKeyTab=true;
};
```

查看 user.keytab 文件，发现 principal 为 zmk_kafka。

图13-5 查看 user.keytab 文件

```
[root@8-5-148-6 client]# klist -kt user.keytab
Keytab name: FILE:user.keytab
KVNO Timestamp         Principal
---- ----------------- --------------------------------------------------------
   1 12/19/16 16:28:17 zmk_kafka@HADOOP.COM
   1 12/19/16 16:28:17 zmk_kafka@HADOOP.COM
```

发现 jaas 文件和 user.keytab 文件中 principal 不对应。

该情况是由于应用程序自动定时更新 Jaas 文件，但是有两个不同的进程在进行更新，一个进程写入正确的 Principal 而另一个却写入了错误的 Principal，以至于程序时而正常，时而异常。

解决办法

步骤 1 修改 Jaas 文件，确保使用的 Principal 在 Keytab 文件中存在。

----结束

13.4 Producer 发送数据失败，抛出 TOPIC_AUTHORIZATION_FAILED

问题现象

使用 MRS 安装集群，主要安装 ZooKeeper、Kafka。

在使用 Producer 向 Kafka 发送数据功能时，发现客户端抛出 TOPIC_AUTHORIZATION_FAILED。

可能原因

1. Kafka 服务异常。
2. 客户端 Producer 侧采用非安全访问，服务端配置禁止访问。

3. 客户端 Producer 侧采用非安全访问，Kafka Topic 设置 ACL。

原因分析

Producer 发送数据到 Kafka 失败，可能原因客户端 Producer 侧问题或者 Kafka 侧问题。

1. 查看 kafka 服务状态：
   - MRS Manager 界面操作：登录 MRS Manager，依次选择 "服务管理 > Kafka ，查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
   - FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka，

     查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
2. 查看 Producer 客户端日志，发现打印 TOPIC_AUTHORIZATION_FAILED 异常信息。

   ```
   [root@10-10-144-2 client]# kafka-console-producer.sh --broker-list
   10.5.144.2:9092 --topic test
   1
   [2017-01-24 16:58:36,671] WARN Error while fetching metadata with correlation
   id 0 : {test=TOPIC_AUTHORIZATION_FAILED}
   (org.apache.kafka.clients.NetworkClient)
   [2017-01-24 16:58:36,672] ERROR Error when sending message to topic test with
   key: null, value: 1 bytes with error: Not authorized to access topics: [test]
   (org.apache.kafka.clients.producer.internals.ErrorLoggingCallback)
   ```

   Producer 采用 9092 端口来访问 Kafka，9092 为非安全端口。
3. 通过 Manager 页面，查看当前 Kafka 集群配置，发现未设置自定义配置“allow.everyone.if.no.acl.found”=“false”。
   - MRS Manager 界面操作入口：登录 MRS Manager，依次选择 “服务管理 > Kafka> 配置”。
   - FusionInsight Manager 界面操作入口：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka > 配置"。
4. 当 acl 设置为 false 不允许采用 9092 来进行访问。
5. 查看 Producer 客户端日志，发现打印 TOPIC_AUTHORIZATION_FAILED 异常信息。

   ```
   [root@10-10-144-2 client]# kafka-console-producer.sh --broker-list
   10.5.144.2:21005 --topic test_acl
   1
   [2017-01-25 11:09:40,012] WARN Error while fetching metadata with correlation
   id 0 : {test_acl=TOPIC_AUTHORIZATION_FAILED}
   (org.apache.kafka.clients.NetworkClient)
   [2017-01-25 11:09:40,013] ERROR Error when sending message to topic test_acl
   with key: null, value: 1 bytes with error: Not authorized to access topics:
   [test_acl] (org.apache.kafka.clients.producer.internals.ErrorLoggingCallback)
   [2017-01-25 11:14:40,010] WARN Error while fetching metadata with correlation
   id 1 : {test_acl=TOPIC_AUTHORIZATION_FAILED}
   (org.apache.kafka.clients.NetworkClient)
   ```

   Producer 采用 21005 端口来访问 Kafka，21005 为非安全端口。
6. 通过客户端命令查看 topic 的 acl 权限设置信息。

```
[root@10-10-144-2 client]# kafka-acls.sh --authorizer-properties
zookeeper.connect=10.5.144.2:24002/kafka --list --topic topic_acl
Current ACLs for resource `Topic:topic_acl`:
 User:test_user has Allow permission for operations: Describe from hosts: *
User:test_user has Allow permission for operations: Write from hosts: *
```

Topic 设置 acl，则不允许采用 9092 来访问。

7. 查看 Producer 客户端日志，发现打印 TOPIC_AUTHORIZATION_FAILED 异常信息。

```
[root@10-10-144-2 client]# kafka-console-producer.sh --broker-list
10.5.144.2:21007 --topic topic_acl --producer.config
/opt/client/Kafka/kafka/config/producer.properties
1
[2017-01-25 12:43:58,506] WARN Error while fetching metadata with correlation
id 0 : {topic acl=TOPIC AUTHORIZATION FAILED}
(org.apache.kafka.clients.NetworkClient)
[2017-01-25 12:43:58,507] ERROR Error when sending message to topic topic acl
with key: null, value: 1 bytes with error: Not authorized to access topics:
[topic_acl] (org.apache.kafka.clients.producer.internals.ErrorLoggingCallback)
```

Producer 采用 21007 端口来访问 Kafka。

8. 通过客户端命令 klist 查询当前认证用户。

```
[root@10-10-144-2 client]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: test@HADOOP.COM

Valid starting     Expires          Service principal
01/25/17 11:06:48  01/26/17 11:06:45  krbtgt/HADOOP.COM@HADOOP.COM
```

当前认证用户为 **test**。

9. 通过客户端命令查看 topic 的 acl 权限设置信息。

```
[root@10-10-144-2 client]# kafka-acls.sh --authorizer-properties
zookeeper.connect=10.5.144.2:2181/kafka --list --topic topic_acl
Current ACLs for resource `Topic:topic_acl`:
 User:test_user has Allow permission for operations: Describe from hosts: *
User:test_user has Allow permission for operations: Write from hosts: *
```

Topic 设置 acl，用户 test_user 具有 producer 权限。test 无权限进行 producer 操作。

解决方法参考步骤 2。

10. 通过 SSH 登录 Kafka Broker：

通过 **cd /var/log/Bigdata/kafka/broker** 命令进入日志目录。

查看 kafka-authorizer.log 发现如下日志提示用户不属于 kafka 或者 kafkaadmin 组。

```
2017-01-25 13:26:33,648 | INFO  | [kafka-request-handler-0] | The principal is
test, belongs to Group : [hadoop, ficommon] | kafka.authorizer.logger
(SimpleAclAuthorizer.scala:169)
2017-01-25 13:26:33,648 | WARN  | [kafka-request-handler-0] | The user is not
belongs to kafka or kafkaadmin group, authorize failed! |
kafka.authorizer.logger (SimpleAclAuthorizer.scala:170)
```

解决方法参考步骤 3。

解决办法

步骤 1 配置自定义配置“allow.everyone.if.no.acl.found”参数为“true”，重启 Kafka 服务。

步骤 2 采用具有权限用户登录。

例如：

kinit test_user

或者赋予用户相关权限。

须知

需要使用 Kafka 管理员用户（属于 kafkaadmin 组）操作。

例如：

kafka-acls.sh --authorizer-properties zookeeper.connect=10.5.144.2:2181/kafka --topic topic_acl --producer --add --allow-principal User:test

```
[root@10-10-144-2 client]# kafka-acls.sh --authorizer-properties
zookeeper.connect=8.5.144.2:2181/kafka --list --topic topic_acl
Current ACLs for resource `Topic:topic_acl`:
 User:test_user has Allow permission for operations: Describe from hosts: *
 User:test_user has Allow permission for operations: Write from hosts: *
  User:test has Allow permission for operations: Describe from hosts: *
  User:test has Allow permission for operations: Write from hosts: *
```

步骤 3 用户加入 **Kafka** 组或者 **Kafkaadmin** 组。

----结束

13.5 Producer 偶现发送数据失败，日志提示 Too many open files in system

问题背景与现象

在使用 Producer 向 Kafka 发送数据功能时，发现客户端发送失败。

图13-6 Producer 发送数据失败

可能原因

1. Kafka 服务异常。
2. 网络异常。
3. Kafka Topic 异常。

原因分析

1. 查看 kafka 服务状态：
   - MRS Manager 界面操作：登录 MRS Manager，依次选择 "服务管理 > Kafka ，查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
   - FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka，

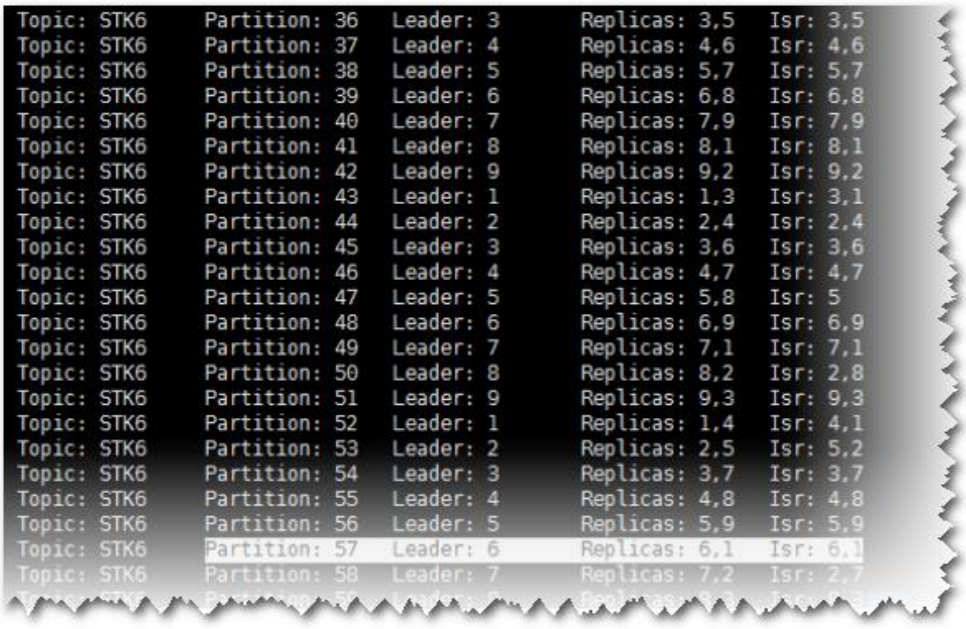
     查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
2. 查看 SparkStreaming 日志中提示错误的 Topic 信息。

   执行 Kafka 相关命令，获取 Topic 分布信息和副本同步信息，观察返回结果。

   kafka-topics.sh --describe --zookeeper <zk_host:port/chroot>

   如图 13-7 所示，发现对应 Topic 状态正常。所有 Partition 均存在正常 Leader 信息。

   图13-7 Topic 状态



3. 通过 telnet 命令，查看是否可以连接 Kafka。

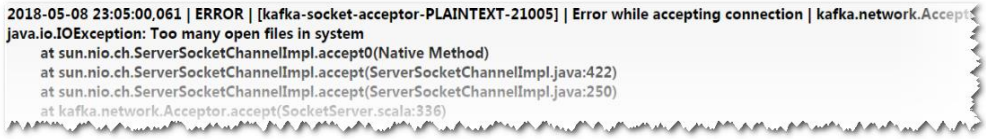
   telnet kafka 业务 ip kafka 业务 port

   如果无法 telnet 成功，请检查网络安全组与 ACL。
4. 通过 SSH 登录 Kafka Broker。

   通过 **cd /var/log/Bigdata/kafka/broker** 命令进入日志目录。

查看 server.log 发现如下日志抛出 java.io.IOException: Too many open files in system。

图13-8 日志异常



5. 通过 lsof 命令查看当前节点 Kafka 进程句柄使用情况，发现占用的句柄数达到了 47 万。

图13-9 句柄数

6. 排查业务代码，不停地 new 新的 producer 对象，未正常关闭。

解决办法

步骤 1 停止当前应用，保证服务端句柄不再疯狂增加影响服务正常运行。

步骤 2 优化应用代码，解决句柄泄露问题。

建议：全局尽量使用一个 Producer 对象，在使用完成之后主动调用 close 接口进行句柄关闭。

----结束

13.6 Consumer 初始化成功，但是无法从 Kafka 中获取指定 Topic 消息

问题背景与现象

使用 MRS 安装集群，主要安装 ZooKeeper、Flume、Kafka、Storm、Spark。

使用 Storm、Spark、Flume 或者自己编写 consumer 代码来消费 Kafka 中指定 Topic 的消息时，发现消费不到任何数据。

可能原因

1. Kafka 服务异常。
2. Consumer 中 ZooKeeper 相关连接地址配置错误，导致无法消费。
3. Consumer 发生 ConsumerRebalanceFailedException 异常，导致无法消费。
4. Consumer 发生网络导致的 ClosedChannelException 异常，导致无法消费。

原因分析

Storm、Spark、Flume 或者自定义 Consumer 代码可以都称为 Consumer。

1. 查看 kafka 服务状态：
   - MRS Manager 界面操作：登录 MRS Manager，依次选择 "服务管理 > Kafka ，查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
   - FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka，

     查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
2. 通过 Kafka Client，判断是否可以正常消费数据。

   假设客户端已经安装在/opt/client 目录，test 为需要消费的 Topic 名称，192.168.234.231 为 ZooKeeper 的 IP 地址。

   ```
   cd /opt/client
   source bigdata_env
   kinit admin
   kafka-topics.sh --zookeeper 192.168.234.231:2181/kafka --describe --topic
   testkafka-console-consumer.sh --topic test --zookeeper
   192.168.234.231:2181/kafka --from-beginning
   ```

   当可以消费到数据时，表示集群服务正常。
3. 查看 Consumer 相关配置，发现 ZooKeeper 连接地址错误。
   - Flume

     ```
     server.sources.Source02.type=org.apache.flume.source.kafka.KafkaSource
     server.sources.Source02.zookeeperConnect=192.168.234.231:2181
     server.sources.Source02.topic = test
     server.sources.Source02.groupId = test_01
     ```
   - Spark

     ```
     val zkQuorum = "192.168.234.231:2181"
     ```
   - Storm

     ```
     BrokerHosts brokerHosts = new ZKHosts("192.168.234.231:2181");
     ```
   - Consumer API

     ```
     zookeeper.connect="192.168.234.231:2181"
     ```

   MRS 中 Kafka 在 ZooKeeper 存储的 ZNode 是以/kafka 为根路径，有别于开源。Kafka 对应的 ZooKeeper 连接配置为 192.168.234.231:2181/kafka。

   Consumer 中配置为 ZooKeeper 连接配置为 192.168.234.231:2181，导致无法正确获取 Kafka 中 Topic 相关信息。

   解决方法参考步骤 1。
4. 查看 Consumer 相关日志，发现打印 ConsumerRebalanceFailedException 异常。

```
2016-02-03 15:55:32,557 | ERROR | [ZkClient-EventThread-75-
192.168.234.231:2181/kafka] |  Error handling event ZkEvent[New session event
sent to
kafka.consumer.ZookeeperConsumerConnector$ZKSessionExpireListener@34b41dfe]  |
org.I0Itec.zkclient.ZkEventThread.run(ZkEventThread.java:77)
kafka.common.ConsumerRebalanceFailedException: pc-zjqbetl86-1454482884879-
2ec95ed3 can't rebalance after 4 retries
at
kafka.consumer.ZookeeperConsumerConnector$ZKRebalancerListener.syncedRebalance(
ZookeeperConsumerConnector.scala:633)
at
kafka.consumer.ZookeeperConsumerConnector$ZKSessionExpireListener.handleNewSess
ion(ZookeeperConsumerConnector.scala:487)
at org.I0Itec.zkclient.ZkClient$4.run(ZkClient.java:472)
at org.I0Itec.zkclient.ZkEventThread.run(ZkEventThread.java:71)
```

通过异常信息，发现当前 Consumer 没有在指定的重试次数内完成 Rebalance，使得 Consumer 没有被分配 Kafka Topic-Partition，则无法消费消息。

解决方法参考步骤 3。

5. 查看 Consumer 相关日志，发现打印 Fetching topic metadata with correlation id 0 for topics [Set(test)] from broker [id:26,host:192-168-234-231,port:9092] failed 错误信息和 ClosedChannelException 异常。

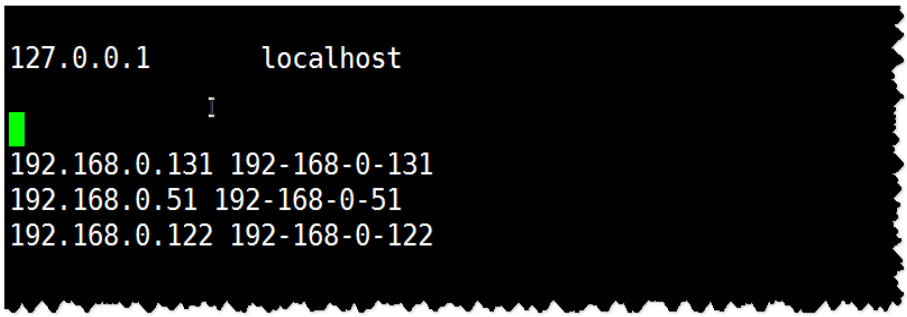
```
[2016-03-04 03:33:53,047] INFO Fetching metadata from broker id:26,host: 192-
168-234-231,port:9092 with correlation id 0 for 1 topic(s) Set(test)
(kafka.client.ClientUtils$)
[2016-03-04 03:33:55,614] INFO Connected to 192-168-234-231:21005 for producing
(kafka.producer.SyncProducer)
[2016-03-04 03:33:55,614] INFO Disconnecting from 192-168-234-231:21005
(kafka.producer.SyncProducer)
[2016-03-04 03:33:55,615] WARN Fetching topic metadata with correlation id 0
for topics [Set(test)] from broker [id:26,host: 192-168-234-231,port:21005]
failed (kafka.client.ClientUtils$)
java.nio.channels.ClosedChannelException
at kafka.network.BlockingChannel.send(BlockingChannel.scala:100)
at kafka.producer.SyncProducer.liftedTree1$1(SyncProducer.scala:73)
at
kafka.producer.SyncProducer.kafka$producer$SyncProducer$$doSend(SyncProducer.sc
ala:72)
at kafka.producer.SyncProducer.send(SyncProducer.scala:113)
at kafka.client.ClientUtils$.fetchTopicMetadata(ClientUtils.scala:58)
at kafka.client.ClientUtils$.fetchTopicMetadata(ClientUtils.scala:93)
at
kafka.consumer.ConsumerFetcherManager$LeaderFinderThread.doWork(ConsumerFetcher
Manager.scala:66)
at kafka.utils.ShutdownableThread.run(ShutdownableThread.scala:60)
[2016-03-04 03:33:55,615] INFO Disconnecting from 192-168-234-231:21005
(kafka.producer.SyncProducer)
```

通过异常信息，发现当前 Consumer 无法从 Kafka Broker 192-168-234-231 节点获取元数据，导致无法连接正确的 Broker 获取消息。

6. 检查网络是否存在问题，如果网络没有问题，检查是否配置主机和 IP 的对应关系
   - Linux

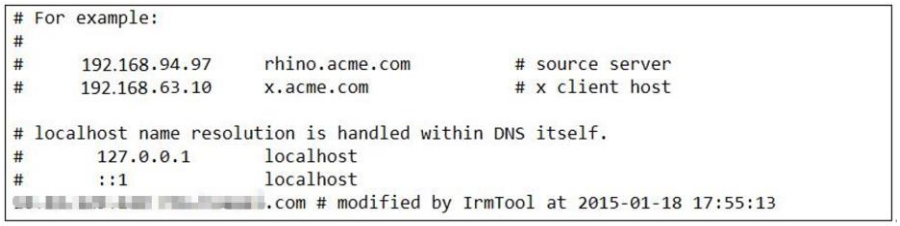
执行 **cat /etc/hosts** 命令。

图13-10 示例 1



- Windows
  打开“C:\Windows\System32\drivers\etc\hosts”。

图13-11 示例 2



解决方法参考步骤 4。

解决办法

步骤 1 ZooKeeper 连接地址配置错误。

步骤 2 修改 Consumer 配置中的 ZooKeeper 连接地址信息，保证和 MRS 相一致。

- Flume

```
server.sources.Source02.type=org.apache.flume.source.kafka.KafkaSource
server.sources.Source02.zookeeperConnect=192.168.234.231:2181/kafka
server.sources.Source02.topic = test
server.sources.Source02.groupId = test_01
```

- Spark

```
val zkQuorum = "192.168.234.231:2181/kafka"
```

- Storm

```
BrokerHosts brokerHosts = new ZKHosts("192.168.234.231:2181/kafka");
```

- Consumer API

```
zookeeper.connect="192.168.234.231:2181/kafka"
```

步骤 3 Rebalance 异常。

同一个消费者组(consumer group)有多个 consumer 先后启动，就是一个消费者组内有多个 consumer 同时消费多个 partition 数据，consumer 端也会有负载均衡（consumer 个数小于 partitions 数量时）。

consumer 实际上是靠存储在 zk 中的临时节点来表明针对哪个 topic 的那个 partition 拥有读权限的。所在路径为：/consumers/consumer-group-xxx/owners/topic-xxx/x。

当触发负载均衡后，原来的 consumer 会重新计算并释放已占用的 partitions，此过程需要一定的处理时间，新来的 consumer 抢占该 partitions 时很有可能会失败。

表13-1 参数说明

名称	作用	默认值
rebalance.max.retries	Rebalance 最大重试次数	4
rebalance.backoff.ms	Rebalance 每次重试间隔	2000
zookeeper.session.timeout.ms	Zookeeper 连接会话超时时间	15000

可以适当调大上述三个参数，可以参考如下数值：

```
zookeeper.session.timeout.ms = 45000
rebalance.max.retries = 10
rebalance.backoff.ms = 5000
```

参数设置应遵循：

rebalance.max.retries * rebalance.backoff.ms > zookeeper.session.timeout.ms

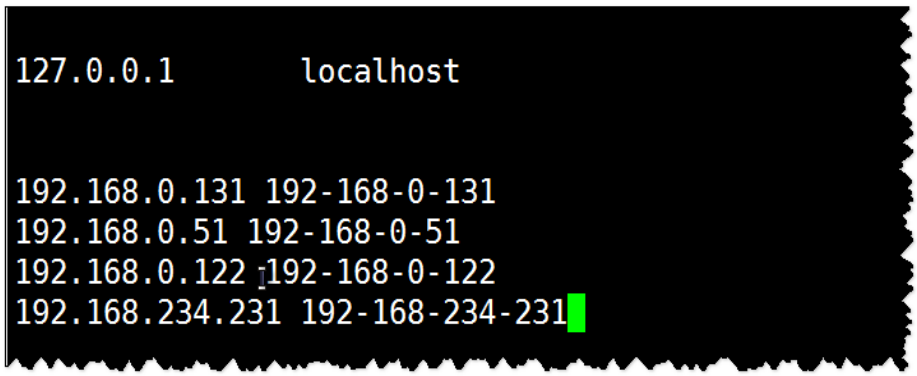
步骤 4 网络异常。

在 hosts 文件中没有配置主机名和 IP 的对应关系，导致使用主机名进行访问时，无法获取信息。

步骤 5 在 hosts 文件中添加对应的主机名和 IP 的对应关系。

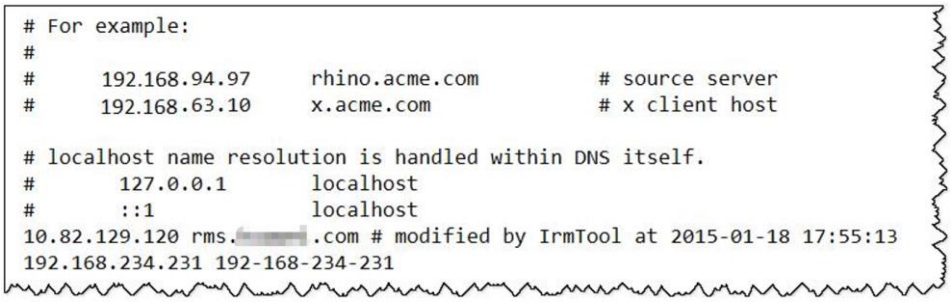
- Linux

图13-12 示例 3



- Windows

图13-13 示例 4



----结束

13.7 Consumer 消费数据失败，Consumer 一直处于等待状态

问题现象

使用 MRS 服务安装集群，主要安装 ZooKeeper、Kafka。

在使用 Consumer 从 Kafka 消费数据时，发现客户端一直处于等待状态。

可能原因

1. Kafka 服务异常。
2. 客户端 Consumer 侧采用非安全访问，服务端配置禁止访问。
3. 客户端 Consumer 侧采用非安全访问，Kafak Topic 设置 ACL。

原因分析

Consumer 向 Kafka 消费数据失败，可能原因客户端 Consumer 侧问题或者 Kafka 侧问题。

1. 查看 kafka 服务状态：
   - MRS Manager 界面操作：登录 MRS Manager，依次选择 "服务管理 > Kafka ，查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
   - FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka，

     查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
2. 查看 Consumer 客户端日志发现频繁连接 Broker 节点和断链打印信息，如下所示。

```
[root@10-10-144-2 client]# kafka-console-consumer.sh --topic test --zookeeper
10.5.144.2:2181/kafka --from-beginning


[2017-03-07 09:22:00,658] INFO Fetching metadata from broker
BrokerEndPoint(1,10.5.144.2,9092) with correlation id 26 for 1 topic(s)
Set(test) (kafka.client.ClientUtils$)
[2017-03-07 09:22:00,659] INFO Connected to 10.5.144.2:9092 for producing
(kafka.producer.SyncProducer)
[2017-03-07 09:22:00,659] INFO Disconnecting from 10.5.144.2:9092
(kafka.producer.SyncProducer)
```

   Consumer 采用 9092 端口来访问 Kafka，9092 为非安全端口。
3. 通过 Manager 页面查看当前 Kafka 集群配置，发现未配置自定义参数“allow.everyone.if.no.acl.found”=“false”。
   - MRS Manager 界面操作入口：登录 MRS Manager，依次选择 “服务管理 > Kafka> 配置”。
   - FusionInsight Manager 界面操作入口：登录 FusionInsight Manager，选择“集群 > *待操作集群的名称* > 服务 > Kafka > 配置"。
4. 当 acl 设置为 false 不允许采用 9092 来进行访问。
5. 查看 Consumer 客户端日志发现频繁连接 Broker 节点和断链打印信息，如下所示。

```
[root@10-10-144-2 client]# kafka-console-consumer.sh --topic test_acl --
zookeeper 10.5.144.2:2181/kafka --from-beginning


[2017-03-07 09:49:16,992] INFO Fetching metadata from broker
BrokerEndPoint(2,10.5.144.3,9092) with correlation id 16 for 1 topic(s)
Set(topic acl) (kafka.client.ClientUtils$)
[2017-03-07 09:49:16,993] INFO Connected to 10.5.144.3:9092 for producing
(kafka.producer.SyncProducer)
[2017-03-07 09:49:16,994] INFO Disconnecting from 10.5.144.3:9092
(kafka.producer.SyncProducer)
```

   Consumer 采用 21005 端口来访问 Kafka，21005 为非安全端口。
6. 通过客户端命令查看 topic 的 acl 权限设置信息。

```
[root@10-10-144-2 client]# kafka-acls.sh --authorizer-properties
zookeeper.connect=10.5.144.2:2181/kafka --list --topic topic_acl
```

```
Current ACLs for resource `Topic:topic_acl`:
 User:test_user has Allow permission for operations: Describe from hosts: *
User:test_user has Allow permission for operations: Write from hosts: *
```

Topic 设置 acl，则不允许采用 9092 来访问。

7. 查看 Consumer 客户端日志发现打印信息，如下所示。

```
[root@10-10-144-2 client]# kafka-console-consumer.sh --topic topic_acl --
bootstrap-server 10.5.144.2:21007 --consumer.config
/opt/client/Kafka/kafka/config/consumer.properties --from-beginning --new-
consumer

[2017-03-07 10:19:18,478] INFO Kafka version : 0.9.0.0
(org.apache.kafka.common.utils.AppInfoParser)
[2017-03-07 10:19:18,478] INFO Kafka commitId : unknown
(org.apache.kafka.common.utils.AppInfoParser)
```

Consumer 采用 21007 端口来访问 Kafka。

8. 通过客户端命令 klist 查询当前认证用户：

```
[root@10-10-144-2 client]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: test@HADOOP.COM

Valid starting     Expires            Service principal
01/25/17 11:06:48  01/26/17 11:06:45  krbtgt/HADOOP.COM@HADOOP.COM
```

当前认证用户为 **test**。

9. 通过客户端命令查看 topic 的 acl 权限设置信息。

```
[root@10-10-144-2 client]# kafka-acls.sh --authorizer-properties
zookeeper.connect=10.5.144.2:24002/kafka --list --topic topic acl
Current ACLs for resource `Topic:topic acl`:
 User:test user has Allow permission for operations: Describe from hosts: *
 User:test user has Allow permission for operations: Write from hosts: *
 User:ttest_user has Allow permission for operations: Read from hosts: *
```

Topic 设置 acl，test 无权限进行 consumer 操作。

解决方法参考步骤 2。

10. 通过 SSH 登录 Kafka Broker：

通过 **cd /var/log/Bigdata/kafka/broker** 命令进入日志目录。

查看 kafka-authorizer.log 发现如下日志提示用户不属于 kafka 或者 kafkaadmin 组。

```
2017-01-25 13:26:33,648 | INFO  | [kafka-request-handler-0] | The principal is
test, belongs to Group : [hadoop, ficommon] | kafka.authorizer.logger
(SimpleAclAuthorizer.scala:169)
2017-01-25 13:26:33,648 | WARN  | [kafka-request-handler-0] | The user is not
belongs to kafka or kafkaadmin group, authorize failed! |
kafka.authorizer.logger (SimpleAclAuthorizer.scala:170)
```

解决方法参考步骤 3。

解决办法

步骤 1 配置自定义参数“allow.everyone.if.no.acl.found”参数为“true”，重启 Kafka 服务。

步骤 2 采用具有权限用户登录。

例如：

kinit test_user

或者赋予用户相关权限。

须知

需要使用 Kafka 管理员用户（属于 kafkaadmin 组）操作。

例如：

kafka-acls.sh --authorizer-properties zookeeper.connect=10.5.144.2:2181/kafka --topic topic_acl --consumer --add --allow-principal User:test --group test

```
[root@10-10-144-2 client]# kafka-acls.sh --authorizer-properties
zookeeper.connect=8.5.144.2:2181/kafka --list --topic topic acl
Current ACLs for resource `Topic:topic acl`:
 User:test user has Allow permission for operations: Describe from hosts: *
 User:test user has Allow permission for operations: Write from hosts: *
  User:test has Allow permission for operations: Describe from hosts: *
  User:test has Allow permission for operations: Write from hosts: *
  User:test has Allow permission for operations: Read from hosts: *
```

步骤 3 用户加入 **Kafka** 组或者 **Kafkaadmin** 组。

----结束

13.8 SparkStreaming 消费 Kafka 消息失败，提示 Error getting partition metadata

问题现象

使用 SparkStreaming 来消费 Kafka 中指定 Topic 的消息时，发现无法从 Kafka 中获取到数据。提示如下错误： Error getting partition metadata。

```
Exception in thread "main" org.apache.spark.SparkException: Error getting
partition metadata for 'testtopic'. Does the topic exist?
org.apache.spark.streaming.kafka.KafkaCluster$$anonfun$checkErrors$1.apply(KafkaClu
ster.scala:366)
org.apache.spark.streaming.kafka.KafkaCluster$$anonfun$checkErrors$1.apply(KafkaClu
ster.scala:366)
scala.util.Either.fold(Either.scala:97)
org.apache.spark.streaming.kafka.KafkaCluster$.checkErrors(KafkaCluster.scala:365)
org.apache.spark.streaming.kafka.KafkaUtils$.createDirectStream(KafkaUtils.scala:42
2)
com.xxx.bigdata.spark.examples.FemaleInfoCollectionPrint$.main(FemaleInfoCollection
Print.scala:45)
com.xxx.bigdata.spark.examples.FemaleInfoCollectionPrint.main(FemaleInfoCollectionP
```

```
rint.scala)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:4
3)
java.lang.reflect.Method.invoke(Method.java:498)
org.apache.spark.deploy.SparkSubmit$.org$apache$spark$deploy$SparkSubmit$$runMain(S
parkSubmit.scala:762)
org.apache.spark.deploy.SparkSubmit$.doRunMain$1(SparkSubmit.scala:183)
org.apache.spark.deploy.SparkSubmit$.submit(SparkSubmit.scala:208)
org.apache.spark.deploy.SparkSubmit$.main(SparkSubmit.scala:123)
org.apache.spark.deploy.SparkSubmit.main(SparkSubmit.scala)
```

可能原因

1. Kafka 服务异常。
2. 客户端 Consumer 侧采用非安全访问，服务端配置禁止访问。
3. 客户端 Consumer 侧采用非安全访问，Kafak Topic 设置 ACL。

原因分析

1. 查看 kafka 服务状态：
   - MRS Manager 界面操作：登录 MRS Manager，依次选择 "服务管理 > Kafka ，查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
   - FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > 待操作集群的名称 > 服务 > Kafka，

     查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
2. 通过 Manager 页面，查看当前 Kafka 集群配置，发现未配置“allow.everyone.if.no.acl.found”或配置为“false”。
   - MRS Manager 界面操作入口：登录 MRS Manager，依次选择 “服务管理 > Kafka> 配置”。
   - FusionInsight Manager 界面操作入口：登录 FusionInsight Manager，选择“集群 > 待操作集群的名称 > 服务 > Kafka > 配置"。
3. 当 acl 设置为 false 不允许采用 Kafka 非安全端口 21005 来进行访问。
4. 通过客户端命令查看 topic 的 acl 权限设置信息：

   ```
   [root@10-10-144-2 client]# kafka-acls.sh --authorizer-properties
   zookeeper.connect=10.5.144.2:2181/kafka --list --topic topic_acl
   Current ACLs for resource `Topic:topic_acl`:
    User:test_user has Allow permission for operations: Describe from hosts: *
   User:test_user has Allow permission for operations: Write from hosts: *
   ```

   Topic 设置 acl，则不允许采用 Kafka 非安全端口 21005 来访问。

解决办法

步骤 1 修改或者添加自定义配置“allow.everyone.if.no.acl.found”参数为“true”，重启 Kafka 服务。

步骤 2 删除 Topic 设置的 ACL。

例如：

kinit test_user

须知

需要使用 Kafka 管理员用户（属于 kafkaadmin 组）操作。

例如：

kafka-acls.sh --authorizer-properties zookeeper.connect=10.5.144.2:2181/kafka --remove --allow-principal User:test_user --producer --topic topic_acl

kafka-acls.sh --authorizer-properties zookeeper.connect=10.5.144.2:2181/kafka --remove --allow-principal User:test_user --consumer --topic topic_acl --group test

----结束

13.9 新建集群 Consumer 消费数据失败，提示 GROUP_COORDINATOR_NOT_AVAILABLE

问题背景与现象

新建 Kafka 集群，部署 Broker 节点数为 2，使用 Kafka 客户端可以正常生产，但是无法正常消费。Consumer 消费数据失败，提示 GROUP_COORDINATOR_NOT_AVAILABLE，关键日志如下：

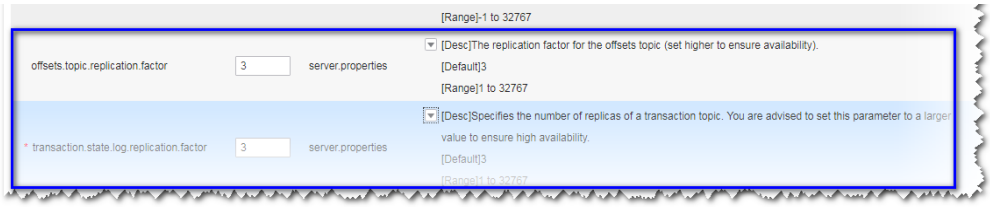
```
2018-05-12 10:58:42,561 | INFO | [kafka-request-handler-3] | [GroupCoordinator 2]:
Preparing to restabilize group DemoConsumer with old generation 118 |
kafka.coordinator.GroupCoordinator (Logging.scala:68)
2018-05-12 10:59:13,562 | INFO | [executor-Heartbeat] | [GroupCoordinator 2]:
Preparing to restabilize group DemoConsumer with old generation 119 |
kafka.coordinator.GroupCoordinator (Logging.scala:68)
```

可能原因

__consumer_offsets 无法创建。

原因分析

1. 查看日志，发现大量__consumer_offset 创建异常。
2. 查看集群发现当前 Broker 数量为 2。
3. 查看__cosumer_offset topic 要求副本为 3，因此创建失败。



解决办法

可以将扩容至至少 3 个流式 core 节点，或参考如下步骤修改服务配置参数。

步骤 1 进入服务参数配置界面。

- MRS Manager 界面操作：登录 MRS Manager，选择“服务管理 > Kafka > 服务配置”，“参数类别”设置为“全部配置”。
- FusionInsight Manager 界面操作：登录 FusionInsight Manager。选择“集群 > 服务 > Kafka”，单击“配置”，选择“全部配置”。

步骤 2 搜索并修改 offsets.topic.replication.factor 和 transaction.state.log.replication.factor 的值为 2。

步骤 3 保存配置，勾选“重新启动受影响的服务或实例。”并单击“确定”重启服务。

----结束

13.10 SparkStreaming 消费 Kafka 消息失败，提示 Couldn't find leader offsets

问题背景与现象

使用 SparkStreaming 来消费 Kafka 中指定 Topic 的消息时，发现无法从 Kafka 中获取到数据。提示如下错误： Couldn't find leader offsets。

```
2018-05-30 12:01:17,816 | INFO  | [Driver] | Reconnect due to socket error: java.net.SocketTimeoutException | kafka.utils.Logging$class.info(Logging.scala:68)
2018-05-30 12:01:47,859 | ERROR | [Driver] | User class threw exception: org.apache.spark.SparkException: java.net.SocketTimeoutException
org.apache.spark.SparkException: Couldn't find leader offsets for Set([STK5,57], [STK5,21]) | org.apache.spark.Logging$class.logError(Logging.scala:96)
org.apache.spark.SparkException: java.net.SocketTimeoutException
org.apache.spark.SparkException: Couldn't find leader offsets for Set([STK5,57], [STK5,21])
        at org.apache.spark.streaming.kafka.KafkaCluster$$anonfun$checkErrors$1.apply(KafkaCluster.scala:366)
        at org.apache.spark.streaming.kafka.KafkaCluster$$anonfun$checkErrors$1.apply(KafkaCluster.scala:366)
        at scala.util.Either.fold(Either.scala:97)
        at org.apache.spark.streaming.kafka.KafkaCluster$.checkErrors(KafkaCluster.scala:365)
        at org.apache.spark.streaming.kafka.KafkaUtils$.createDirectStream(KafkaUtils.scala:422)
        at org.apache.spark.streaming.kafka.KafkaUtils$.createDirectStream(KafkaUtils.scala:532)
        at org.apache.spark.streaming.kafka.KafkaUtils.createDirectStream(KafkaUtils.scala)
        at com.stk.bigdata.sparkstreaming.notify.SparkAlarmControlwz.main(SparkAlarmControlwz.java:194)
        at com.stk.bigdata.sparkstreaming.submit.SparkNotifyA.main(SparkNotifyA.java:14)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.spark.deploy.yarn.ApplicationMaster$$anon$2.run(ApplicationMaster.scala:540)
2018-05-30 12:01:47,863 | INFO  | [Driver] | Final app status: FAILED, exitCode: 15, (reason: User class threw exception: org.apache.spark.SparkException: java.
org.apache.spark.SparkException: Couldn't find leader offsets for Set([STK5,57], [STK5,21])) | org.apache.spark.Logging$class.logInfo(Logging.scala:59)
2018-05-30 12:01:47,866 | INFO  | [pool-1-thread-1] | Invoking stop() from shutdown hook | org.apache.spark.Logging$class.logInfo(Logging.scala:59)
```

可能原因

- Kafka 服务异常。
- 网络异常。

- Kafka Topic 异常。

原因分析

步骤 1 通过 Manager 页面，查看 Kafka 集群当前状态，发现状态为“良好”，且监控指标内容显示正确。

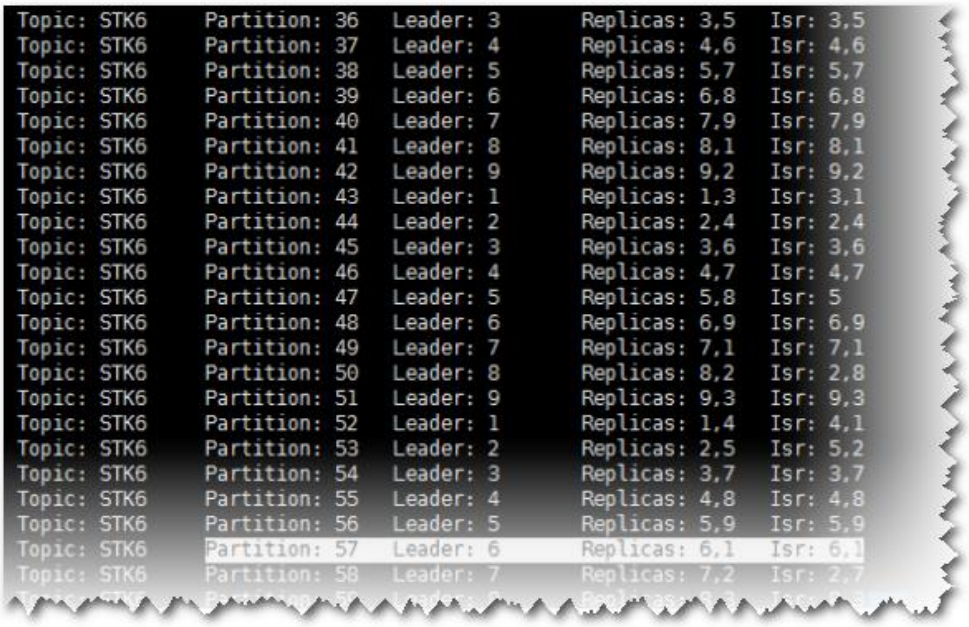
步骤 2 查看 SparkStreaming 日志中提示错误的 Topic 信息。

执行 Kafka 相关命令，获取 Topic 分布信息和副本同步信息，观察返回结果。

kafka-topics.sh --describe --zookeeper <zk_host:port/chroot> --topic <topic name>

如下所示，发现对应 Topic 状态正常。所有 Partition 均存在正常 Leader 信息。

图13-14 Topic 分布信息和副本同步信息



步骤 3 检查客户端与 Kafka 集群网络是否连通，若网络不通协调网络组进行处理。

步骤 4 通过 SSH 登录 Kafka Broker。

通过 **cd /var/log/Bigdata/kafka/broker** 命令进入日志目录。

查看 server.log 发现如下日志抛出 java.lang.OutOfMemoryError: Direct buffer memory。

```
2018-05-30 12:02:00,246 | ERROR | [kafka-network-thread-6-PLAINTEXT-3] | Processor
got uncaught exception. | kafka.network.Processor (Logging.scala:103)

 java.lang.OutOfMemoryError: Direct buffer memory
 at java.nio.Bits.reserveMemory(Bits.java:694)
 at java.nio.DirectByteBuffer.<init>(DirectByteBuffer.java:123)
 at java.nio.ByteBuffer.allocateDirect(ByteBuffer.java:311)
```

```
at sun.nio.ch.Util.getTemporaryDirectBuffer(Util.java:241)
at sun.nio.ch.IOUtil.read(IOUtil.java:195)
at sun.nio.ch.SocketChannelImpl.read(SocketChannelImpl.java:380)
```

at org.apache.kafka.common.network.PlaintextTransportLayer.read(PlaintextTransportLayer.java:110)

步骤 5 通过 Manager 页面，查看当前 Kafka 集群配置。

- MRS Manager 界面操作：登录 MRS Manager，选择“服务管理 > Kafka > 服务配置”，“参数类别”设置为“全部配置”，发现“KAFKA_JVM_PERFORMANCE_OPTS”的中“-XX:MaxDirectMemorySize”值为“1G”。
- FusionInsight Manager 界面操作：登录 FusionInsight Manager。选择“集群 > 服务 > Kafka”，单击“配置”，选择“全部配置”，发现“KAFKA_JVM_PERFORMANCE_OPTS”的中“-XX:MaxDirectMemorySize”值为“1G”。

步骤 6 直接内存配置过小导致报错，而且一旦直接内存溢出，该节点将无法处理新请求，会导致其他节点或者客户端访问超时失败。

----结束

解决办法

步骤 1 登录到 Manager，进入 Kafka 配置页面。

- MRS Manager 界面操作：登录 MRS Manager，选择“服务管理 > Kafka > 服务配置”。
- FusionInsight Manager 界面操作：登录 FusionInsight Manager。选择“集群 > 服务 > Kafka”，单击“配置”。

步骤 2 选择“全部配置”，搜索并修改 KAFKA_JVM_PERFORMANCE_OPTS 的值。

步骤 3 保存配置，勾选“重新启动受影响的服务或实例。”并单击“确定”重启服务。

----结束

13.11 Consumer 消费数据失败，提示 SchemaException: Error reading field 'brokers'

问题背景与现象

Consumer 来消费 Kafka 中指定 Topic 的消息时，发现无法从 Kafka 中获取到数据。提示如下错误： org.apache.kafka.common.protocol.types.SchemaException: Error reading field 'brokers': Error reading field 'host': Error reading string of length 28271, only 593 bytes available。

```
Exception in thread "Thread-0"
org.apache.kafka.common.protocol.types.SchemaException: Error reading field
'brokers': Error reading field 'host': Error reading string of length 28271, only
593 bytes available
at org.apache.kafka.common.protocol.types.Schema.read(Schema.java:73)
at org.apache.kafka.clients.NetworkClient.parseResponse(NetworkClient.java:380)
at
org.apache.kafka.clients.NetworkClient.handleCompletedReceives(NetworkClient.java:4
49)
at org.apache.kafka.clients.NetworkClient.poll(NetworkClient.java:269)
at
org.apache.kafka.clients.consumer.internals.ConsumerNetworkClient.clientPoll(Consum
erNetworkClient.java:360)
at
org.apache.kafka.clients.consumer.internals.ConsumerNetworkClient.poll(ConsumerNetw
orkClient.java:224)
at
org.apache.kafka.clients.consumer.internals.ConsumerNetworkClient.poll(ConsumerNetw
orkClient.java:192)
at
org.apache.kafka.clients.consumer.internals.ConsumerNetworkClient.poll(ConsumerNetw
orkClient.java:163)
atorg.apache.kafka.clients.consumer.internals.AbstractCoordinator.ensureCoordinator
Ready(AbstractCoordinator.java:179)
at org.apache.kafka.clients.consumer.KafkaConsumer.pollOnce(KafkaConsumer.java:973)
at org.apache.kafka.clients.consumer.KafkaConsumer.poll(KafkaConsumer.java:937)
at KafkaNew.Consumer$ConsumerThread.run(Consumer.java:40)
```

可能原因

客户端和服务端 Jar 版本不一致。

解决办法

修改 Consumer 应用程序中 kafka jar，确保和服务端保持一致。

13.12 Consumer 消费数据是否丢失排查

问题背景与现象

客户将消费完的数据存入数据库，发现数据与生产数据不一致，怀疑 Kafka 消费丢数据

可能原因

- 客户代码原因
- Kafka 生产数据写入异常
- Kafka 消费数据异常

解决办法

Kafka 排查：

步骤 1 通过 consumer-groups.sh 来观察写入和消费的 offerset 的变化情况（生产一定数量的消息，客户端进行消费，观察 offerset 的变化）。

步骤 2 新建一个消费组，用客户端进行消费，然后查看消费的消息。

new-consumer:

kafka-console-consumer.sh --topic <topic name> --bootstrap-server <IP1:PORT, IP2:PORT,...> --new-consumer --consumer.config <config file>

----结束

客户代码排查：

步骤 1 查看客户端里有没有提交 offerset 的报错。

步骤 2 如果没有报错把消费的 API 里加上打印消息，打印少量数据（只打印 key 即可），查看丢失的数据。

----结束

13.13 帐号锁定导致启动组件失败

问题背景与现象

新安装集群，启动 Kafka 失败。显示认证失败，导致启动失败。

```
/home/omm/kerberos/bin/kinit -k -t ${BIGDATA_HOME}/etc/2_15_ Broker /kafka.keytab
kafka/hadoop.hadoop.com -c ${BIGDATA_HOME}/etc/2_15_ Broker /11846 failed.
export key tab file for kafka/hadoop.hadoop.com failed.export and check keytab file
failed, errMsg=]}] for Broker #192.168.1.92@192-168-1-92.
[2015-07-11 02:34:33] RoleInstance started failure for ROLE[name: Broker].
[2015-07-11 02:34:34] Failed to complete the instances start operation. Current
operation entities: [Broker #192.168.1.92@192-168-1-92], Failure entites : [Broker
#192.168.1.92@192-168-1-92].Operation Failed.Failed to complete the instances start
operation. Current operation entities: [Broker#192.168.1.92@192-168-1-92], Failure
entites: [Broker #192.168.1.92@192-168-1-92].
```

原因分析

查看 Kerberos 日志，/var/log/Bigdata/kerberos/krb5kdc.log，发现有集群外的 IP 使用 kafka 用户连接，导致多次认证失败，最终导致 Kafka 帐号被锁定。

```
Jul 11 02:49:16 192-168-1-91 krb5kdc[1863](info): AS REQ (2 etypes {18 17})
192.168.1.93: NEEDED PREAUTH: kafka/hadoop.hadoop.com@HADOOP.COM for
krbtgt/HADOOP.COM@HADOOP.COM, Additional pre-authentication required
Jul 11 02:49:16 192-168-1-91 krb5kdc[1863](info): preauth (encrypted timestamp)
verify failure: Decrypt integrity check failed
Jul 11 02:49:16 192-168-1-91 krb5kdc[1863](info): AS REQ (2 etypes {18 17})
192.168.1.93: PREAUTH FAILED: kafka/hadoop.hadoop.com@HADOOP.COM for
krbtgt/HADOOP.COM@HADOOP.COM, Decrypt integrity check failed
```

解决办法

进入集群外的节点（如原因分析示例中的 192.168.1.93），断开其对 Kafka 的认证。等待 5 分钟，此帐号就会被解锁。

13.14 Kafka Broker 上报进程异常，日志提示 IllegalArgumentException

问题背景与现象

使用 Manager 提示进程故障告警，查看告警进程为 Kafka Broker。

可能原因

Broker 配置异常。

原因分析

1. 在 Manager 页面，在告警页面得到主机信息。
2. 通过 SSH 登录 Kafka Broker，执行 **cd /var/log/Bigdata/kafka/broker** 命令进入日志目录。

   查看 server.log 发现如下日志抛出 IllegalArgumentException 异常，提示 java.lang.IllegalArgumentException: requirement failed: replica.fetch.max.bytes should be equal or greater than message.max.bytes。

   ```
   2017-01-25 09:09:14,930 | FATAL | [main] |  | kafka.Kafka$ (Logging.scala:113)
   java.lang.IllegalArgumentException: requirement failed: replica.fetch.max.bytes
   should be equal or greater than message.max.bytes
         at scala.Predef$.require(Predef.scala:233)
         at kafka.server.KafkaConfig.validateValues(KafkaConfig.scala:959)
         at kafka.server.KafkaConfig.<init>(KafkaConfig.scala:944)
         at kafka.server.KafkaConfig$.fromProps(KafkaConfig.scala:701)
         at kafka.server.KafkaConfig$.fromProps(KafkaConfig.scala:698)
         at
   kafka.server.KafkaServerStartable$.fromProps(KafkaServerStartable.scala:28)
         at kafka.Kafka$.main(Kafka.scala:60)
         at kafka.Kafka.main(Kafka.scala)
   ```

   Kafka 要求 replica.fetch.max.bytes 需要大于等于 message.max.bytes。

3. 进入 Kafka 配置页面，选择“全部配置”， 显示所有 Kafka 相关配置，分别搜索 message.max.bytes、

   replica.fetch.max.bytes 进行检索，发现 replica.fetch.max.bytes 小于 message.max.bytes。

解决办法

步骤 1 登录 Manager 界面，进入 Kafka 配置页面。

- MRS 3.x 之前的版本：登录 MRS Manager，选择“服务管理 > Kafka > 配置 > 全部配置”。
- MRS 3.x 及后续版本，登录 FusionInsight Manager，选择“集群 > 服务 > Kafka > 配置 > 全部配置”。

步骤 2 搜索并修改 replica.fetch.max.bytes 参数，使得 replica.fetch.max.bytes 的值大于等于 message.max.bytes，使得不同 Broker 上的 Partition 的 Replica 可以同步到全部消息。

步骤 3 保存配置，查看集群是否存在配置过期的服务，如果存在，需重启对应服务或角色实例使配置生效。

步骤 4 修改消费者业务应用中 fetch.message.max.bytes，使得 fetch.message.max.bytes 的值大于等于 message.max.bytes，确保消费者可以消费到全部消息。

----结束

13.15 执行 Kafka Topic 删除操作，发现无法删除

问题背景与现象

在使用 Kafka 客户端命令删除 Topic 时，发现 Topic 无法被删除。

```
kafka-topics.sh --delete --topic test --zookeeper 192.168.234.231:2181/kafka
```

可能原因

- 客户端命令连接 ZooKeeper 地址错误。
- Kafka 服务异常 Kafka 部分节点处于停止状态。
- Kafka 服务端配置禁止删除。
- Kafka 配置自动创建，且 Producer 未停止。

原因分析

1. 客户端命令，打印 ZkTimeoutException 异常。

```
[2016-03-09 10:41:45,773] WARN Can not get the principle name from server
192.168.234.231 (org.apache.zookeeper.ClientCnxn)
Exception in thread "main" org.I0Itec.zkclient.exception.ZkTimeoutException:
Unable to connect to zookeeper server within timeout: 30000
at org.I0Itec.zkclient.ZkClient.connect(ZkClient.java:880)
at org.I0Itec.zkclient.ZkClient.<init>(ZkClient.java:98)
```

```
at org.I0Itec.zkclient.ZkClient.<init>(ZkClient.java:84)
at kafka.admin.TopicCommand$.main(TopicCommand.scala:51)
at kafka.admin.TopicCommand.main(TopicCommand.scala)
```

解决方法参考步骤 1。

2. 客户端查询命令。

   kafka-topics.sh --list --zookeeper 192.168.0.122:2181/kafka

   ```
   test - marked for deletion
   ```

   通过 Manager 查看 Kafka Broker 实例的运行状态。

   通过 **cd /var/log/Bigdata/kafka/broker** 命令进入 RunningAsController 节点日志目录，在 controller.log 发现 ineligible for deletion: test。

   ```
   2016-03-09 11:11:26,228 | INFO  | [main] | [Controller 1]: List of topics to be deleted:  | kafka.controller.KafkaController (Logging.scala:68)
   2016-03-09 11:11:26,229 | INFO  | [main] | [Controller 1]: List of topics ineligible for deletion: test | kafka.controller.KafkaController (Logging.scala:68)
   ```

3. 通过 Manager 查询 Broker 删除 Topic 相关配置。

   解决方法参考步骤 2

4. 客户端查询命令：

   ```
   kafka-topics.sh --describe -topic test --zookeeper 192.168.0.122:2181/kafka
   ```

   ```
   Topic:test      PartitionCount:10       ReplicationFactor:2     Configs:
           Topic: test     Partition: 0    Leader: -1      Replicas: 1,2   Isr:
           Topic: test     Partition: 1    Leader: -1      Replicas: 2,3   Isr:
           Topic: test     Partition: 2    Leader: -1      Replicas: 3,1   Isr:
           Topic: test     Partition: 3    Leader: -1      Replicas: 1,3   Isr:
           Topic: test     Partition: 4    Leader: -1      Replicas: 2,1   Isr:
           Topic: test     Partition: 5    Leader: -1      Replicas: 3,2   Isr:
           Topic: test     Partition: 6    Leader: -1      Replicas: 1,2   Isr:
           Topic: test     Partition: 7    Leader: -1      Replicas: 2,3   Isr:
           Topic: test     Partition: 8    Leader: -1      Replicas: 3,1   Isr:
           Topic: test     Partition: 9    Leader: -1      Replicas: 1,3   Isr:
   ```

   进入 RunningAsController 节点日志目录，在 controller.log 发现 marked ineligible for deletion。

   ```
   2016-03-10 11:11:17,989 | INFO  | [delete-topics-thread-3] | [delete-topics-thread-3], Handling deletion for topics test | kafka.controller.TopicDeletionManager$DeleteTopicsThread (Logging.scala:68)
   2016-03-10 11:11:17,990 | INFO  | [delete-topics-thread-3] | [delete-topics-thread-3], Not retrying deletion of topic test at this time since it is marked ineligible for deletion | kafka.controller.TopicDeletionManager$DeleteTopicsThread (Logging.scala:68)
   ```

5. 通过 Manager 查询 Broker 状态。

   其中一个 Broker 处于停止或者故障状态。Topic 进行删除必须保证该 Topic 的所有 Partition 所在的 Broker 必须处于正常状态。

   解决方法参考步骤 3。

6. 进入 RunningAsController 节点日志目录，在 controller.log 发现 Deletion successfully，然后又出现 New topics: [Set(test)]，表明被再次创建。

   ```
   2016-03-10 11:33:35,208 | INFO  | [delete-topics-thread-3] | [delete-topics-thread-3], Deletion of topic test successfully completed | kafka.controller.TopicDeletionManager$DeleteTopicsThread (Logging.scala:68)
   ```

```
2016-03-10 11:33:38,501 | INFO  | [ZkClient-EventThread-19-
192.168.0.122:2181,160.172.0.52:2181,160.172.0.51:2181/kafka] |
[TopicChangeListener on Controller 3]: New topics: [Set(test)], deleted topics:
[Set()], new partition replica assignment
```

7. 通过 Manager 查询 Broker 创建 Topic 相关配置。

   经确认，对该 Topic 操作的应用没有停止。

   解决方法参考步骤 4。

解决办法

步骤 1 ZooKeeper 连接失败导致。

Kafka 客户端连接 ZooKeeper 服务超时。检查客户端到 ZooKeeper 的网络连通性。

网络连接失败，通过 Manager 界面查看 Zookeeper 服务信息。

配置错误，修改客户端命令中 ZooKeeper 地址。

步骤 2 Kafka 服务端配置禁止删除。

通过 Manager 界面修改 delete.topic.enable 为 true。保存配置并重启服务。

客户端查询命令，无 Topic:test。

```
kafka-topics.sh --list --zookeeper 192.168.0.122:24002/kafka
```

进入 RunningAsController 节点日志目录，在 controller.log 发现 Deletion of topic test successfully。

```
2016-03-10 10:39:40,665 | INFO  | [delete-topics-thread-3] | [Partition state
machine on Controller 3]: Invoking state change to OfflinePartition for partitions
[test,2],[test,15],[test,6],[test,16],[test,12],[test,7],[test,10],[test,13],[test,
9],[test,19],[test,3],[test,5],[test,1],[test,0],[test,17],[test,8],[test,4],[test,
11],[test,14],[test,18] | kafka.controller.PartitionStateMachine (Logging.scala:68)
2016-03-10 10:39:40,668 | INFO  | [delete-topics-thread-3] | [Partition state
machine on Controller 3]: Invoking state change to NonExistentPartition for
partitions
[test,2],[test,15],[test,6],[test,16],[test,12],[test,7],[test,10],[test,13],[test,
9],[test,19],[test,3],[test,5],[test,1],[test,0],[test,17],[test,8],[test,4],[test,
11],[test,14],[test,18] | kafka.controller.PartitionStateMachine (Logging.scala:68)
2016-03-10 10:39:40,977 | INFO  | [delete-topics-thread-3] | [delete-topics-thread-
3], Deletion of topic test successfully completed |
kafka.controller.TopicDeletionManager$DeleteTopicsThread (Logging.scala:68)
```

步骤 3 Kafka 部分节点处于停止或者故障状态。

启动停止的 Broker 实例。

客户端查询命令，无 Topic:test。

```
kafka-topics.sh --list --zookeeper 192.168.0.122:24002/kafka
```

进入 RunningAsController 节点日志目录，在 controller.log 发现 Deletion of topic test successfully。

```
2016-03-10 11:17:56,463 | INFO  | [delete-topics-thread-3] | [Partition state
machine on Controller 3]: Invoking state change to NonExistentPartition for
partitions
[test,4],[test,1],[test,8],[test,2],[test,5],[test,9],[test,7],[test,6],[test,0],[t
est,3] | kafka.controller.PartitionStateMachine (Logging.scala:68)
2016-03-10 11:17:56,726 | INFO  | [delete-topics-thread-3] | [delete-topics-thread-
3], Deletion of topic test successfully completed |
kafka.controller.TopicDeletionManager$DeleteTopicsThread (Logging.scala:68)
```

步骤 4 Kafka 配置自动创建，且 Producer 未停止。

停止相关应用，通过 Manager 界面修改“auto.create.topics.enable”为“false”，保存配置并重启服务。

步骤 5 再次执行 delete 操作。

----结束

13.16 执行 Kafka Topic 删除操作，提示 AdminOperationException

问题背景与现象

在使用 Kafka 客户端命令设置 Topic ACL 权限时，发现 Topic 无法被设置。

```
kafka-topics.sh --delete --topic test4 --zookeeper 10.5.144.2:2181/kafka
```

提示错误 ERROR kafka.admin.AdminOperationException: Error while deleting topic test4。

具体如下：

```
Error while executing topic command : Error while deleting topic test4
[2017-01-25 14:00:20,750] ERROR kafka.admin.AdminOperationException: Error while
deleting topic test4
at kafka.admin.TopicCommand$$anonfun$deleteTopic$1.apply(TopicCommand.scala:177)
at kafka.admin.TopicCommand$$anonfun$deleteTopic$1.apply(TopicCommand.scala:162)
at scala.collection.mutable.ResizableArray$class.foreach(ResizableArray.scala:59)
at scala.collection.mutable.ArrayBuffer.foreach(ArrayBuffer.scala:47)
at kafka.admin.TopicCommand$.deleteTopic(TopicCommand.scala:162)
at kafka.admin.TopicCommand$.main(TopicCommand.scala:68)
at kafka.admin.TopicCommand.main(TopicCommand.scala)
 (kafka.admin.TopicCommand$)
```

可能原因

用户不属于 **kafkaadmin** 组，Kafka 提供安全访问接口，kafkaamdin 组用户才可以进行 topic 删除操作。

原因分析

1. 使用客户端命令，打印 AdminOperationException 异常。
2. 通过客户端命令 **klist** 查询当前认证用户：

```
[root@10-10-144-2 client]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: test@HADOOP.COM

Valid starting     Expires           Service principal
01/25/17 11:06:48  01/26/17 11:06:45  krbtgt/HADOOP.COM@HADOOP.COM
```

如上例中当前认证用户为 test。

3. 通过命令 **id** 查询用户组信息

```
[root@10-10-144-2 client]# id test
uid=20032(test) gid=10001(hadoop)
groups=10001(hadoop),9998(ficommon),10003(kafka)
```

解决办法

MRS Manager 界面操作：

步骤 1 登录 MRS Manager。

步骤 2 选择“系统设置 > 用户管理”。

步骤 3 在操作用户对应的“操作”列，单击“修改”。

步骤 4 为用户加入 **kafkaadmin** 组。单击“确定”完成修改操作。

步骤 5 通过命令 **id** 查询用户组信息。

```
[root@10-10-144-2 client]# id test
uid=20032(test) gid=10001(hadoop)
groups=10001(hadoop),9998(ficommon),10002(kafkaadmin), 10003(kafka)
```

----结束

FusionInsight Manager 界面操作：

步骤 1 登录 FusionInsight Manager。

步骤 2 选择“系统 > 权限 > 用户”。

步骤 3 在使用的用户所在行的单击“修改”。

步骤 4 为用户添加 **kafkaadmin** 组。单击“确定”完成修改操作。

步骤 5 通过命令 **id** 查询用户组信息。

```
[root@10-10-144-2 client]# id test
uid=20032(test) gid=10001(hadoop)
groups=10001(hadoop),9998(ficommon),10002(kafkaadmin), 10003(kafka)
```

----结束

13.17 执行 Kafka Topic 创建操作，发现无法创建提示 NoAuthException

问题背景与现象

在使用 Kafka 客户端命令创建 Topic 时，发现 Topic 无法被创建。

```
kafka-topics.sh --create --zookeeper 192.168.234.231:2181/kafka --replication-factor 1 --partitions 2 --topic test
```

提示错误 NoAuthException，KeeperErrorCode = NoAuth for /config/topics。

具体如下：

```
Error while executing topic command
org.apache.zookeeper.KeeperException$NoAuthException: KeeperErrorCode = NoAuth for /config/topics
org.I0Itec.zkclient.exception.ZkException:
org.apache.zookeeper.KeeperException$NoAuthException: KeeperErrorCode = NoAuth for /config/topics
 at org.I0Itec.zkclient.exception.ZkException.create(ZkException.java:68)
 at org.I0Itec.zkclient.ZkClient.retryUntilConnected(ZkClient.java:685)
 at org.I0Itec.zkclient.ZkClient.create(ZkClient.java:304)
 at org.I0Itec.zkclient.ZkClient.createPersistent(ZkClient.java:213)
 at kafka.utils.ZkUtils$.createParentPath(ZkUtils.scala:215)
 at kafka.utils.ZkUtils$.updatePersistentPath(ZkUtils.scala:338)
 at kafka.admin.AdminUtils$.writeTopicConfig(AdminUtils.scala:247)
```

可能原因

用户不属于 **kafkaadmin** 组，Kafka 提供安全访问接口，kafkaamdin 组用户才可以进行 topic 删除操作。

原因分析

1. 使用客户端命令，打印 NoAuthException 异常。

   ```
   Error while executing topic command
   org.apache.zookeeper.KeeperException$NoAuthException: KeeperErrorCode = NoAuth for /config/topics
   org.I0Itec.zkclient.exception.ZkException:
   org.apache.zookeeper.KeeperException$NoAuthException: KeeperErrorCode = NoAuth for /config/topics
    at org.I0Itec.zkclient.exception.ZkException.create(ZkException.java:68)
    at org.I0Itec.zkclient.ZkClient.retryUntilConnected(ZkClient.java:685)
    at org.I0Itec.zkclient.ZkClient.create(ZkClient.java:304)
    at org.I0Itec.zkclient.ZkClient.createPersistent(ZkClient.java:213)
    at kafka.utils.ZkUtils$.createParentPath(ZkUtils.scala:215)
    at kafka.utils.ZkUtils$.updatePersistentPath(ZkUtils.scala:338)
    at kafka.admin.AdminUtils$.writeTopicConfig(AdminUtils.scala:247)
   ```

2. 通过客户端命令 **klist** 查询当前认证用户：

```
[root@10-10-144-2 client]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: test@HADOOP.COM

Valid starting     Expires          Service principal
01/25/17 11:06:48  01/26/17 11:06:45  krbtgt/HADOOP.COM@HADOOP.COM
```

如上例中当前认证用户为 **test**。

3. 通过命令 **id** 查询用户组信息。

```
[root@10-10-144-2 client]# id test
uid=20032(test) gid=10001(hadoop)
groups=10001(hadoop),9998(ficommon),10003(kafka)
```

解决办法

MRS Manager 界面操作：

步骤 1 登录 MRS Manager。

步骤 2 选择“系统设置 > 用户管理”。

步骤 3 在操作用户对应的“操作”列，单击“修改”。

步骤 4 为用户加入 **kafkaadmin** 组。

步骤 5 通过命令 **id** 查询用户组信息。

```
[root@10-10-144-2 client]# id test
uid=20032(test) gid=10001(hadoop)
groups=10001(hadoop),9998(ficommon),10002(kafkaadmin), 10003(kafka)
```

----结束

FusionInsight Manager 界面操作：

步骤 1 登录 FusionInsight Manager。

步骤 2 选择“系统 > 权限 > 用户”。

步骤 3 在使用的用户所在行的单击“修改”。

步骤 4 为用户添加 **kafkaadmin** 组。单击“确定”完成修改操作。

步骤 5 通过命令 **id** 查询用户组信息。

```
[root@10-10-144-2 client]# id test
uid=20032(test) gid=10001(hadoop)
groups=10001(hadoop),9998(ficommon),10002(kafkaadmin), 10003(kafka)
```

----结束

13.18 执行 Kafka Topic 设置 ACL 操作失败，提示 NoAuthException

问题背景与现象

在使用 Kafka 客户端命令设置 Topic ACL 权限时，发现 Topic 无法被设置。

```
kafka-acls.sh --authorizer-properties zookeeper.connect=10.5.144.2:2181/kafka  --
topic topic_acl --producer --add --allow-principal User:test_acl
```

提示错误 NoAuthException: KeeperErrorCode = NoAuth for /kafka-acl-changes/acl_changes_0000000002。

具体如下：

```
Error while executing ACL command:
org.apache.zookeeper.KeeperException$NoAuthException: KeeperErrorCode = NoAuth for
/kafka-acl-changes/acl_changes_0000000002
org.I0Itec.zkclient.exception.ZkException:
org.apache.zookeeper.KeeperException$NoAuthException: KeeperErrorCode = NoAuth for
/kafka-acl-changes/acl_changes_0000000002
at org.I0Itec.zkclient.exception.ZkException.create(ZkException.java:68)
at org.I0Itec.zkclient.ZkClient.retryUntilConnected(ZkClient.java:995)
at org.I0Itec.zkclient.ZkClient.delete(ZkClient.java:1038)
at kafka.utils.ZkUtils.deletePath(ZkUtils.scala:499)
at
kafka.common.ZkNodeChangeNotificationListener$$anonfun$purgeObsoleteNotifications$1
.apply(ZkNodeChangeNotificationListener.scala:118)
at
kafka.common.ZkNodeChangeNotificationListener$$anonfun$purgeObsoleteNotifications$1
.apply(ZkNodeChangeNotificationListener.scala:112)
at scala.collection.mutable.ResizableArray$class.foreach(ResizableArray.scala:59)
at scala.collection.mutable.ArrayBuffer.foreach(ArrayBuffer.scala:47)
at
kafka.common.ZkNodeChangeNotificationListener.purgeObsoleteNotifications(ZkNodeChan
geNotificationListener.scala:112)
at
kafka.common.ZkNodeChangeNotificationListener.kafka$common$ZkNodeChangeNotification
Listener$$processNotifications(ZkNodeChangeNotificationListener.scala:97)
at
kafka.common.ZkNodeChangeNotificationListener.processAllNotifications(ZkNodeChangeN
otificationListener.scala:77)
at
kafka.common.ZkNodeChangeNotificationListener.init(ZkNodeChangeNotificationListener
.scala:65)
at kafka.security.auth.SimpleAclAuthorizer.configure(SimpleAclAuthorizer.scala:136)
at kafka.admin.AclCommand$.withAuthorizer(AclCommand.scala:73)
at kafka.admin.AclCommand$.addAcl(AclCommand.scala:80)
at kafka.admin.AclCommand$.main(AclCommand.scala:48)
at kafka.admin.AclCommand.main(AclCommand.scala)
Caused by: org.apache.zookeeper.KeeperException$NoAuthException: KeeperErrorCode =
NoAuth for /kafka-acl-changes/acl_changes_0000000002
at org.apache.zookeeper.KeeperException.create(KeeperException.java:117)
```

```
at org.apache.zookeeper.KeeperException.create(KeeperException.java:51)
at org.apache.zookeeper.ZooKeeper.delete(ZooKeeper.java:1416)
at org.I0Itec.zkclient.ZkConnection.delete(ZkConnection.java:104)
at org.I0Itec.zkclient.ZkClient$11.call(ZkClient.java:1042)
at org.I0Itec.zkclient.ZkClient.retryUntilConnected(ZkClient.java:985)
```

可能原因

用户不属于 **kafkaadmin** 组，Kafka 提供安全访问接口，**kafkaamdin** 组用户才可以进行设置操作。

原因分析

1. 使用客户端命令，打印 NoAuthException 异常。
2. 通过客户端命令 **klist** 查询当前认证用户：

```
[root@10-10-144-2 client]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: test@HADOOP.COM

Valid starting     Expires            Service principal
01/25/17 11:06:48  01/26/17 11:06:45  krbtgt/HADOOP.COM@HADOOP.COM
```

如上例中当前认证用户为 **test**。

3. 通过命令 **id** 查询用户组信息。

```
[root@10-10-144-2 client]# id test
uid=20032(test) gid=10001(hadoop)
groups=10001(hadoop),9998(ficommon),10003(kafka)
```

解决办法

MRS Manager 界面操作：

步骤 1 登录 MRS Manager。

步骤 2 选择“系统设置 > 用户管理”。

步骤 3 在操作用户对应的“操作”列，单击“修改”。

步骤 4 为用户加入 **kafkaadmin** 组。

步骤 5 通过命令 **id** 查询用户组信息。

```
[root@host1 client]# id test
uid=20032(test) gid=10001(hadoop)
groups=10001(hadoop),9998(ficommon),10002(kafkaadmin), 10003(kafka)
```

----结束

FusionInsight Manager 界面操作：

步骤 1 登录 FusionInsight Manager。

步骤 2 选择“系统 > 权限 > 用户”。

步骤 3 在使用的用户所在行的单击“修改”。

步骤 4 为用户添加 **kafkaadmin** 组。单击“确定”完成修改操作。

步骤 5 通过命令 **id** 查询用户组信息。

```
[root@10-10-144-2 client]# id test
uid=20032(test) gid=10001(hadoop)
groups=10001(hadoop),9998(ficommon),10002(kafkaadmin), 10003(kafka)
```

----结束

13.19 执行 Kafka Topic 创建操作，发现无法创建提示 NoNode for /brokers/ids

问题背景与现象

在使用 Kafka 客户端命令创建 Topic 时，发现 Topic 无法被创建。

```
kafka-topics.sh --create  --replication-factor 1 --partitions 2 --topic test --
zookeeper 192.168.234.231:2181
```

提示错误 NoNodeException: KeeperErrorCode = NoNode for /brokers/ids。

具体如下：

```
Error while executing topic command :
org.apache.zookeeper.KeeperException$NoNodeException: KeeperErrorCode = NoNode for
/brokers/ids
[2017-09-17 16:35:28,520] ERROR org.I0Itec.zkclient.exception.ZkNoNodeException:
org.apache.zookeeper.KeeperException$NoNodeException: KeeperErrorCode = NoNode for
/brokers/ids
    at org.I0Itec.zkclient.exception.ZkException.create(ZkException.java:47)
    at org.I0Itec.zkclient.ZkClient.retryUntilConnected(ZkClient.java:995)
    at org.I0Itec.zkclient.ZkClient.getChildren(ZkClient.java:675)
    at org.I0Itec.zkclient.ZkClient.getChildren(ZkClient.java:671)
    at kafka.utils.ZkUtils.getChildren(ZkUtils.scala:541)
    at kafka.utils.ZkUtils.getSortedBrokerList(ZkUtils.scala:176)
    at kafka.admin.AdminUtils$.createTopic(AdminUtils.scala:235)
    at kafka.admin.TopicCommand$.createTopic(TopicCommand.scala:105)
    at kafka.admin.TopicCommand$.main(TopicCommand.scala:60)
    at kafka.admin.TopicCommand.main(TopicCommand.scala)
Caused by: org.apache.zookeeper.KeeperException$NoNodeException: KeeperErrorCode =
NoNode for /brokers/ids
    at org.apache.zookeeper.KeeperException.create(KeeperException.java:115)
    at org.apache.zookeeper.KeeperException.create(KeeperException.java:51)
    at org.apache.zookeeper.ZooKeeper.getChildren(ZooKeeper.java:2256)
    at org.apache.zookeeper.ZooKeeper.getChildren(ZooKeeper.java:2284)
    at org.I0Itec.zkclient.ZkConnection.getChildren(ZkConnection.java:114)
    at org.I0Itec.zkclient.ZkClient$4.call(ZkClient.java:678)
    at org.I0Itec.zkclient.ZkClient$4.call(ZkClient.java:675)
    at org.I0Itec.zkclient.ZkClient.retryUntilConnected(ZkClient.java:985)
```

```
    ... 8 more
(kafka.admin.TopicCommand$)
```

可能原因

- Kafka 服务处于停止状态。
- 客户端命令中 zookeeper 地址参数配置错误。

原因分析

1. 使用客户端命令，打印 NoNodeException 异常。

```
Error while executing topic command :
org.apache.zookeeper.KeeperException$NoNodeException: KeeperErrorCode = NoNode
for /brokers/ids
[2017-09-17 16:35:28,520] ERROR org.I0Itec.zkclient.exception.ZkNoNodeException:
org.apache.zookeeper.KeeperException$NoNodeException: KeeperErrorCode = NoNode
for /brokers/ids
    at org.I0Itec.zkclient.exception.ZkException.create(ZkException.java:47)
    at org.I0Itec.zkclient.ZkClient.retryUntilConnected(ZkClient.java:995)
    at org.I0Itec.zkclient.ZkClient.getChildren(ZkClient.java:675)
    at org.I0Itec.zkclient.ZkClient.getChildren(ZkClient.java:671)
    at kafka.utils.ZkUtils.getChildren(ZkUtils.scala:541)
    at kafka.utils.ZkUtils.getSortedBrokerList(ZkUtils.scala:176)
    at kafka.admin.AdminUtils$.createTopic(AdminUtils.scala:235)
    at kafka.admin.TopicCommand$.createTopic(TopicCommand.scala:105)
    at kafka.admin.TopicCommand$.main(TopicCommand.scala:60)
    at kafka.admin.TopicCommand.main(TopicCommand.scala)
```

2. 通过 Manager 查看 Kafka 服务是否处于正常状态。
3. 检查客户端命令中 ZooKeeper 地址是否正确，访问 ZooKeeper 上所存放的 Kafka 信息，其路径（Znode）应该加上/kafka，发现配置中缺少/kafka：

```
[root@10-10-144-2 client]#
kafka-topics.sh --create  --replication-factor 1 --partitions 2 --topic test --
zookeeper 192.168.234.231:2181
```

解决办法

步骤 1 保证 Kafka 服务处于正常状态。

步骤 2 创建命令中 ZooKeeper 地址信息需要添加/kafka。

```
[root@10-10-144-2 client]#
kafka-topics.sh --create  --replication-factor 1 --partitions 2 --topic test --
zookeeper 192.168.234.231:2181/kafka
```

----结束

13.20 执行 Kakfa Topic 创建操作，发现无法创建提示 replication factor larger than available brokers

问题背景与现象

在使用 Kafka 客户端命令创建 Topic 时，发现 Topic 无法被创建。

```
kafka-topics.sh --create  --replication-factor 2 --partitions 2 --topic test --
zookeeper 192.168.234.231:2181
```

提示错误 replication factor larger than available brokers。

具体如下：

```
Error while executing topic command : replication factor: 2 larger than available
brokers: 0
[2017-09-17 16:44:12,396] ERROR kafka.admin.AdminOperationException: replication
factor: 2 larger than available brokers: 0
    at kafka.admin.AdminUtils$.assignReplicasToBrokers(AdminUtils.scala:117)
    at kafka.admin.AdminUtils$.createTopic(AdminUtils.scala:403)
    at kafka.admin.TopicCommand$.createTopic(TopicCommand.scala:110)
    at kafka.admin.TopicCommand$.main(TopicCommand.scala:61)
    at kafka.admin.TopicCommand.main(TopicCommand.scala)
 (kafka.admin.TopicCommand$)
```

可能原因

- Kafka 服务处于停止状态。
- Kafka 服务当前可用 Broker 小于设置的 replication-factor。
- 客户端命令中 Zookeeper 地址参数配置错误。

原因分析

1. 使用客户端命令，打印 replication factor larger than available brokers 异常。

   ```
   Error while executing topic command : replication factor: 2 larger than
   available brokers: 0
   [2017-09-17 16:44:12,396] ERROR kafka.admin.AdminOperationException:
   replication factor: 2 larger than available brokers: 0
       at kafka.admin.AdminUtils$.assignReplicasToBrokers(AdminUtils.scala:117)
       at kafka.admin.AdminUtils$.createTopic(AdminUtils.scala:403)
       at kafka.admin.TopicCommand$.createTopic(TopicCommand.scala:110)
       at kafka.admin.TopicCommand$.main(TopicCommand.scala:61)
       at kafka.admin.TopicCommand.main(TopicCommand.scala)
    (kafka.admin.TopicCommand$)
   ```

2. 通过 Manager 参看 Kafka 服务是否处于正常状态，当前可用 Broker 是否小于设置的 replication-factor。
3. 检查客户端命令中 ZooKeeper 地址是否正确，访问 ZooKeeper 上所存放的 Kafka 信息，其路径（Znode）应该加上/kafka，发现配置中缺少/kafka。

```
[root@10-10-144-2 client]#
kafka-topics.sh --create  --replication-factor 2 --partitions 2 --topic test --
zookeeper 192.168.234.231:2181
```

解决办法

步骤 1　保证 Kafka 服务处于正常状态，且可用 Broker 不小于设置的 replication-factor。

步骤 2　创建命令中 ZooKeeper 地址信息需要添加/kafka。

```
[root@10-10-144-2 client]#
kafka-topics.sh --create  --replication-factor 1 --partitions 2 --topic test --
zookeeper 192.168.234.231:2181/kafka
```

----结束

13.21 Consumer 消费数据存在重复消费现象

问题背景与现象

当数据量较大时会频繁的发生 rebalance 导致出现重复消费的情况，关键日志如下：

```
2018-05-12 10:58:42,561 | INFO | [kafka-request-handler-3] | [GroupCoordinator 2]:
Preparing to restabilize group DemoConsumer with old generation 118 |
kafka.coordinator.GroupCoordinator (Logging.scala:68)
2018-05-12 10:58:43,245 | INFO | [kafka-request-handler-5] | [GroupCoordinator 2]:
Stabilized group DemoConsumer generation 119 | kafka.coordinator.GroupCoordinator
(Logging.scala:68)
2018-05-12 10:58:43,560 | INFO | [kafka-request-handler-7] | [GroupCoordinator 2]:
Assignment received from leader for group DemoConsumer for generation 119 |
kafka.coordinator.GroupCoordinator (Logging.scala:68)
2018-05-12 10:59:13,562 | INFO | [executor-Heartbeat] | [GroupCoordinator 2]:
Preparing to restabilize group DemoConsumer with old generation 119 |
kafka.coordinator.GroupCoordinator (Logging.scala:68)
2018-05-12 10:59:13,790 | INFO | [kafka-request-handler-3] | [GroupCoordinator 2]:
Stabilized group DemoConsumer generation 120 | kafka.coordinator.GroupCoordinator
(Logging.scala:68)
2018-05-12 10:59:13,791 | INFO | [kafka-request-handler-0] | [GroupCoordinator 2]:
Assignment received from leader for group DemoConsumer for generation 120 |
kafka.coordinator.GroupCoordinator (Logging.scala:68)
2018-05-12 10:59:43,802 | INFO | [kafka-request-handler-2] | Rolled new log segment
for '__consumer_offsets-17' in 2 ms. | kafka.log.Log (Logging.scala:68)
2018-05-12 10:59:52,456 | INFO | [group-metadata-manager-0] | [Group Metadata
Manager on Broker 2]: Removed 0 expired offsets in 0 milliseconds. |
kafka.coordinator.GroupMetadataManager (Logging.scala:68)
2018-05-12 11:00:49,772 | INFO | [kafka-scheduler-6] | Deleting segment 0 from log
__consumer_offsets-17. | kafka.log.Log (Logging.scala:68)
2018-05-12 11:00:49,773 | INFO | [kafka-scheduler-6] | Deleting index
/srv/BigData/kafka/data4/kafka-logs/  consumer offsets-
17/00000000000000000000.index.deleted | kafka.log.OffsetIndex (Logging.scala:68)
2018-05-12 11:00:49,773 | INFO | [kafka-scheduler-2] | Deleting segment 2147948547
from log   consumer offsets-17. | kafka.log.Log (Logging.scala:68)
2018-05-12 11:00:49,773 | INFO | [kafka-scheduler-4] | Deleting segment 4282404355
```

```
from log   consumer offsets-17. | kafka.log.Log (Logging.scala:68)
2018-05-12 11:00:49,775 | INFO | [kafka-scheduler-2] | Deleting index
/srv/BigData/kafka/data4/kafka-logs/  consumer offsets-
17/00000000002147948547.index.deleted | kafka.log.OffsetIndex (Logging.scala:68)
2018-05-12 11:00:49,775 | INFO | [kafka-scheduler-4] | Deleting index
/srv/BigData/kafka/data4/kafka-logs/  consumer offsets-
17/00000000004282404355.index.deleted | kafka.log.OffsetIndex (Logging.scala:68)
2018-05-12 11:00:50,533 | INFO | [kafka-scheduler-6] | Deleting segment 4283544095
from log   consumer offsets-17. | kafka.log.Log (Logging.scala:68)
2018-05-12 11:00:50,569 | INFO | [kafka-scheduler-6] | Deleting index
/srv/BigData/kafka/data4/kafka-logs/__consumer_offsets-
17/00000000004283544095.index.deleted | kafka.log.OffsetIndex (Logging.scala:68)
2018-05-12 11:02:21,178 | INFO | [kafka-request-handler-2] | [GroupCoordinator 2]:
Preparing to restabilize group DemoConsumer with old generation 120 |
kafka.coordinator.GroupCoordinator (Logging.scala:68)
2018-05-12 11:02:22,839 | INFO | [kafka-request-handler-4] | [GroupCoordinator 2]:
Stabilized group DemoConsumer generation 121 | kafka.coordinator.GroupCoordinator
(Logging.scala:68)
2018-05-12 11:02:23,169 | INFO | [kafka-request-handler-1] | [GroupCoordinator 2]:
Assignment received from leader for group DemoConsumer for generation 121 |
kafka.coordinator.GroupCoordinator (Logging.scala:68)
2018-05-12 11:02:49,913 | INFO | [kafka-request-handler-6] | Rolled new log segment
for '__consumer_offsets-17' in 2 ms. | kafka.log.Log (Logging.scala:68)
```

其中 Preparing to restabilize group DemoConsumer with old generation 表示正在发生 rebalance。

可能原因

参数设置不合理。

原因分析

原因：由于参数设置不当，数据量大时数据处理时间过长，导致频繁发生 balance，此时 offset 无法正常提交，导致重复消费数据。

原理：每次 poll 的数据处理完后才提交 offset，如果 poll 数据后的处理时长超出了 session.timeout.ms 的设置时长，此时发生 rebalance 导致本次消费失败，已经消费数据的 offset 无法正常提交，所以下次重新消费时还是在旧的 offset 消费数据，从而导致消费数据重复。

解决办法

建议用户在 Manager 页面调整以下服务参数：

request.timeout.ms=100000

session.timeout.ms=90000

max.poll.records=50

heartbeat.interval.ms=3000

其中：

request.timeout.ms 要比 session.timeout.ms 大 10s。

session.timeout.ms 的大小设置要在服务端参数 group.min.session.timeout.ms 和 group.max.session.timeout.ms 之间。

以上参数可以根据实际情况进行适当的调整，特别是 max.poll.records，这个参数是为了控制每次 poll 数据的 records 量，保证每次的处理时长尽量保持稳定。目的是为了保证 poll 数据以后的处理时间不要超过 session.timeout.ms 的时间。

参考信息

- poll 之后的数据处理效率要高，不要阻塞下一次 poll
- poll 方法和数据处理建议异步处理

13.22 执行 Kafka Topic 创建操作，发现 Partition 的 Leader 显示为 none

问题背景与现象

在使用 Kafka 客户端命令创建 Topic 时，发现创建 Topic Partition 的 Leader 显示为 none。

```
[root@10-10-144-2 client]#
kafka-topics.sh --create  --replication-factor 1 --partitions 2 --topic test --
zookeeper 10.6.92.36:2181/kafka

Created topic "test".
[root@10-10-144-2 client]#
kafka-topics.sh --describe --zookeeper 10.6.92.36:2181/kafka

Topic:test     PartitionCount:2       ReplicationFactor:2    Configs:
       Topic: test    Partition: 0    Leader: none      Replicas: 2,3   Isr:
       Topic: test    Partition: 1    Leader: none      Replicas: 3,1   Isr:
```

可能原因

- Kafka 服务处于停止状态。
- 找不到用户组信息。

原因分析

1. 查看 kafka 服务状态及监控指标。
   - MRS Manager 界面操作：登录 MRS Manager，依次选择 "服务管理 > Kafka ，查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
   - FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > 待操作集群的名称 > 服务 > Kafka，查看当前 Kafka 状态，发现状态为良好，且监控指标内容显示正确。
2. 在 Kafka 概览页面获取 Controller 节点信息。

3. 登录 Controller 所在节点，通过 **cd /var/log/Bigdata/kafka/broker** 命令进入节点日志目录，在 state-change.log 发现存在 ZooKeeper 权限异常，提示 NoAuthException。

```
2018-05-31 09:20:42,436 | ERROR | [ZkClient-EventThread-34-
10.6.92.36:24002,10.6.92.37:24002,10.6.92.38:24002/kafka] | Controller 4 epoch
6 initiated state change for partition [test,1] from NewPartition to
OnlinePartition failed | state.change.logger (Logging.scala:103)

 org.I0Itec.zkclient.exception.ZkException:
org.apache.zookeeper.KeeperException$NoAuthException: KeeperErrorCode = NoAuth
for /brokers/topics/test/partitions
 at org.I0Itec.zkclient.exception.ZkException.create(ZkException.java:68)
 at org.I0Itec.zkclient.ZkClient.retryUntilConnected(ZkClient.java:1000)
 at org.I0Itec.zkclient.ZkClient.create(ZkClient.java:527)
 at org.I0Itec.zkclient.ZkClient.createPersistent(ZkClient.java:293)
```

4. 查看对应时间段 ZooKeeper 审计日志，权限异常。

```
2018-05-31 09:20:42,421 | ERROR | CommitProcWorkThread-1 |
session=0xc3000007015d5a18
    user=10.6.92.39,kafka/hadoop.hadoop.com@HADOOP.COM,kafka/hadoop.hadoop.com
@HADOOP.COM  ip=10.6.92.39operation=create znodetarget=ZooKeeperServer
    znode=/kafka/brokers/topics/test/partitions/0/state  result=failure
2018-05-31 09:20:42,423 | ERROR | CommitProcWorkThread-1 |
session=0xc3000007015d5a18
    user=10.6.92.39,kafka/hadoop.hadoop.com@HADOOP.COM,kafka/hadoop.hadoop.com
@HADOOP.COM  ip=10.6.92.39operation=create znodetarget=ZooKeeperServer
    znode=/kafka/brokers/topics/test/partitions/0    result=failure
2018-05-31 09:20:42,435 | ERROR | CommitProcWorkThread-1 |
session=0xc3000007015d5a18
    user=10.6.92.39,kafka/hadoop.hadoop.com@HADOOP.COM,kafka/hadoop.hadoop.com
@HADOOP.COM  ip=10.6.92.39operation=create znodetarget=ZooKeeperServer
    znode=/kafka/brokers/topics/test/partitions result=failure
2018-05-31 09:20:42,439 | ERROR | CommitProcWorkThread-1 |
session=0xc3000007015d5a18
    user=10.6.92.39,kafka/hadoop.hadoop.com@HADOOP.COM,kafka/hadoop.hadoop.com
@HADOOP.COM  ip=10.6.92.39operation=create znodetarget=ZooKeeperServer
    znode=/kafka/brokers/topics/test/partitions/1/state  result=failure
2018-05-31 09:20:42,441 | ERROR | CommitProcWorkThread-1 |
session=0xc3000007015d5a18
    user=10.6.92.39,kafka/hadoop.hadoop.com@HADOOP.COM,kafka/hadoop.hadoop.com
@HADOOP.COM  ip=10.6.92.39operation=create znodetarget=ZooKeeperServer
    znode=/kafka/brokers/topics/test/partitions/1    result=failure
2018-05-31 09:20:42,453 | ERROR | CommitProcWorkThread-1 |
session=0xc3000007015d5a18
    user=10.6.92.39,kafka/hadoop.hadoop.com@HADOOP.COM,kafka/hadoop.hadoop.com
@HADOOP.COM  ip=10.6.92.39operation=create znodetarget=ZooKeeperServer
    znode=/kafka/brokers/topics/test/partitions result=failure
```

5. 在 ZooKeeper 各个实例节点上执行 **id -Gn kafka** 命令，发现有一个节点无法查询用户组信息。

```
[root @bdpsit3ap03 ~]# id -Gn kafka
id:  kafka: No such user
[root @bdpsit3ap03 ~]#
```

6. MRS 集群中的用户管理由 LDAP 服务管理提供，又依赖于操作系统的 sssd（red hat），nscd（suse）服务，用户的建立到同步到 sssd 服务需要一定时间，如果此时用户没有生效，或者 sssd 版本存在 bug 的情况下，某些情况下在 ZooKeeper 节点会出现用户无效的情况，导致创建 Topic 异常。

解决办法

步骤 1 重启 sssd/nscd 服务。

- Red Hat

```
service sssd restart
```

- SUSE

```
sevice nscd restart
```

步骤 2 重启相关服务后，在节点通过 **id username** 命令查看相应用户信息是否已有效。

----结束

13.23 Kafka 安全使用说明

Kafka API 简单说明

- 新 Producer API

  指 org.apache.kafka.clients.producer.KafkaProducer 中定义的接口，在使用“kafka-console-producer.sh”时，默认使用此 API。

- 旧 Producer API

  指 kafka.producer.Producer 中定义的接口，在使用“kafka-console-producer.sh”时，加“--old-producer”参数会调用此 API。

- 新 Consumer API

  指 org.apache.kafka.clients.consumer.KafkaConsumer 中定义的接口，在使用“kafka-console-consumer.sh”时，加“--new-consumer”参数会调用此 API。

- 旧 Consumer API

  指 kafka.consumer.ConsumerConnector 中定义的接口，在使用“kafka-console-consumer.sh”时，默认使用此 API。

说明

新 Producer API 和新 Consumer API，在下文中统称为新 API。

Kafka 访问协议说明

Kafka 当前支持四种协议类型的访问：PLAINTEXT、SSL、SASL_PLAINTEXT、SASL_SSL。

Kafka 服务启动时，默认会启动 PLAINTEXT 和 SASL_PLAINTEXT 两种协议类型的访问监听。可通过设置 Kafka 服务配置参数“ssl.mode.enable”为“true”，来启动 SSL 和 SASL_SSL 两种协议类型的访问监听。

下表是四中协议类型的简单说明：

协议类型	说明	支持的 API	默认端口
PLAINTEXT	支持无认证的明文访问	新 API 和旧 API	9092
SASL_PLAINTEXT	支持 Kerberos 认证的明文访问	新 API	21007
SSL	支持无认证的 SSL 加密访问	新 API	9093
SASL_SSL	支持 Kerberos 认证的 SSL 加密访问	新 API	21009

Topic 的 ACL 设置

Kafka 支持安全访问，因此可以针对 Topic 进行 ACL 设置，从而控制不同的用户可以访问不同的 Topic。Topic 的权限信息，需要在 Linux 客户端上，使用“kafka-acls.sh”脚本进行查看和设置。

- 操作场景

  该任务指导 Kafka 管理员根据业务需求，为其他使用 Kafka 的系统用户授予相关 Topic 的特定权限。

  Kafka 默认用户组信息表所示。

用户组名	描述
kafkaadmin	Kafka 管理员用户组。添加入本组的用户，拥有所有 Topic 的创建，删除，授权及读写权限。
kafkasuperuser	添加入本组的用户，拥有所有 Topic 的读写权限。
kafka	Kafka 普通用户组。添加入本组的用户，需要被 kafkaadmin 组用户授予特定 Topic 的读写权限，才能访问对应 Topic。

- 前提条件

  a. 系统管理员已明确业务需求，并准备一个 Kafka 管理员用户（属于 kafkaadmin 组）。

  b. 已安装 Kafka 客户端。

- 操作步骤

  a. 以客户端安装用户，登录安装 Kafka 客户端的节点。

  b. 切换到 Kafka 客户端安装目录，例如“/opt/kafkaclient”。

cd /opt/kafkaclient

c. 执行以下命令，配置环境变量。

source bigdata_env

d. 执行以下命令，进行用户认证。（普通集群跳过此步骤）

kinit *组件业务用户*

e. 执行以下命令，切换到 Kafka 客户端安装目录。

cd Kafka/kafka/bin

f. 使用“kafka-acl.sh”进行用户授权常用命令如下：

- 查看某 Topic 权限控制列表：

  ./kafka-acls.sh --authorizer-properties zookeeper.connect=<ZooKeeper 集群业务 IP:2181/kafka > --list --topic <Topic 名称>

- 添加给某用户 Producer 权限：

  ./kafka-acls.sh --authorizer-properties zookeeper.connect=<ZooKeeper 集群业务 IP:2181/kafka > --add --allow-principal User:<用户名> --producer --topic <Topic 名称>

- 删除某用户 Producer 权限：

  ./kafka-acls.sh --authorizer-properties zookeeper.connect=<ZooKeeper 集群业务 IP:2181/kafka > --remove --allow-principal User:<用户名> --producer --topic <Topic 名称>

- 添加给某用户 Consumer 权限：

  ./kafka-acls.sh --authorizer-properties zookeeper.connect=<ZooKeeper 集群业务 IP:2181/kafka > --add --allow-principal User:<用户名> --consumer --topic <Topic 名称> --group <消费者组名称>

- 删除某用户 Consumer 权限：

  ./kafka-acls.sh --authorizer-properties zookeeper.connect=<ZooKeeper 集群业务 IP:2181/kafka > --remove --allow-principal User:<用户名> --consumer --topic <Topic 名称> --group <消费者组名称>

针对不同的 Topic 访问场景 ，Kafka 新旧 API 使用说明

- 场景一：访问设置了 ACL 的 Topic

使用的 API	用户属组	客户端参数	服务端参数	访问的端口
新 API	用户需满足以下条件之一即可： • 属于系统管理员组 • 属于 kafkaadmin 组 • 属于 kafkasuperu	security.protocol=SASL_PLAINTEXT sasl.kerberos.service.name = kafka	-	sasl.port（默认 21007）
		security.protocol=SASL_SSL sasl.kerberos.service.name = kafka	ssl.mode.enable 配置为 true	sasl-ssl.port（默认 21009）

使用的 API	用户属组	客户端参数	服务端参数	访问的端口
	ser 组 • 被授权的 kafka 组的用户			
旧 API	不涉及	不涉及	不涉及	不涉及

- 场景二：访问未设置 ACL 的 Topic

使用的 API	用户属组	客户端参数	服务端参数	访问的端口
新 API	用户需满足以下条件之一： • 属于系统管理员组 • 属于 kafkaadmin 组 • 属于 kafkasuperuser 组	security.protocol=SASL_PLAINTEXT sasl.kerberos.service.name = kafka	-	sasl.port（默认 21007）
	用户属于 kafka 组		allow.everyone.if.no.acl.found 配置为 true	sasl.port（默认 21007）
	用户需满足以下条件之一： • 属于系统管理员组 • 属于 kafkaadmin 组 • kafkasuperuser 组用户	security.protocol=SASL_SSL sasl.kerberos.service.name = kafka	ssl-enable 配置为“true”	sasl-ssl.port（默认 21009）
	用户属于 kafka 组		allow.everyone.if.no.acl.found 配置为“true” ssl-enable 配置为“true”	sasl-ssl.port（默认 21009）
	-	security.protocol=PLAINTEXT	allow.everyone.if.no.acl.found 配置为“true”	port（默认 21005）

使用的 API	用户属组	客户端参数	服务端参数	访问的端口
	-	security.protocol=SSL	allow.everyone.if.no.acl.found 配置为“true” ssl-enable 配置为“true”	ssl.port（默认 21008）
旧 Producer	-	-	allow.everyone.if.no.acl.found 配置为“true”	port（默认 21005）
旧 Consumer	-	-	allow.everyone.if.no.acl.found 配置为“true”	ZooKeeper 服务端口：clientPort（默认 24002）

13.24 如何获取 Kafka Consumer Offset 信息

问题背景与现象

使用 Kafka Consumer 消费数据时，如何获取 Kafka Consumer Offset 相关信息？

Kafka API 简单说明

- 新 Producer API

  指 org.apache.kafka.clients.producer.KafkaProducer 中定义的接口，在使用“kafka-console-producer.sh”时，默认使用此 API。

- 旧 Producer API

  指 kafka.producer.Producer 中定义的接口，在使用“kafka-console-producer.sh”时，加“--old-producer”参数会调用此 API。

- 新 Consumer API

  指 org.apache.kafka.clients.consumer.KafkaConsumer 中定义的接口，在使用“kafka-console-consumer.sh”时，加“--new-consumer”参数会调用此 API。

- 旧 Consumer API

  指 kafka.consumer.ConsumerConnector 中定义的接口，在使用“kafka-console-consumer.sh”时，默认使用此 API。

说明

新 Producer API 和新 Consumer API，在下文中统称为新 API。

处理步骤

旧 Consumer API

- 前提条件
  a. 系统管理员已明确业务需求，并准备一个 Kafka 管理员用户（属于 kafkaadmin 组）。
  b. 已安装 Kafka 客户端。
- 操作步骤
  a. 以客户端安装用户，登录安装 Kafka 客户端的节点。
  b. 切换到 Kafka 客户端安装目录，例如“/opt/kafkaclient”。
     cd /opt/kafkaclient
  c. 执行以下命令，配置环境变量。
     source bigdata_env
  d. 执行以下命令，进行用户认证。（普通模式跳过此步骤）
     kinit *组件业务用户*
  e. 执行以下命令，切换到 Kafka 客户端安装目录。
     cd Kafka/kafka/bin
  f. 执行以下命令，获取 consumer offset metric 信息。

```
 bin/kafka-consumer-groups.sh --zookeeper <zookeeper_host:port>/kafka --list
bin/kafka-consumer-groups.sh --zookeeper <zookeeper_host:port>/kafka --describe --group test-consumer-group
```

例如：

```
kafka-consumer-groups.sh --zookeeper 192.168.100.100:2181/kafka --list
kafka-consumer-groups.sh --zookeeper 192.168.100.100:2181/kafka --describe --group test-consumer-group
```

新 Consumer API

- 前提条件
  a. 系统管理员已明确业务需求，并准备一个 Kafka 管理员用户（属于 kafkaadmin 组）。
  b. 已安装 Kafka 客户端。
- 操作步骤
  a. 以客户端安装用户，登录安装 Kafka 客户端的节点。
  b. 切换到 Kafka 客户端安装目录，例如“/opt/client”。
     cd /opt/client
  c. 执行以下命令，配置环境变量。
     source bigdata_env
  d. 执行以下命令，进行用户认证。（普通模式跳过此步骤）
     kinit *组件业务用户*
  e. 执行以下命令，切换到 Kafka 客户端安装目录。
     cd Kafka/kafka/bin
  f. 执行以下命令，获取 consumer offset metric 信息。

kafka-consumer-groups.sh --bootstrap-server <broker_host:port> --describe --group my-group

例如：

kafka-consumer-groups.sh --bootstrap-server 192.168.100.100:9092 --describe --group my-group

13.25 如何针对 Topic 进行配置增加和删除

问题背景与现象

使用 Kafka 过程中常常需要对特定 Topic 进行配置或者修改。

Topic 级别可以修改参数列表：

```
cleanup.policy
compression.type
delete.retention.ms
file.delete.delay.ms
flush.messages
flush.ms
index.interval.bytes
max.message.bytes
min.cleanable.dirty.ratio
min.insync.replicas
preallocate
retention.bytes
retention.ms
segment.bytes
segment.index.bytes
segment.jitter.ms
segment.ms
unclean.leader.election.enable
```

处理步骤

- 前提条件

  已安装 Kafka 客户端。

- 操作步骤

  a. 以客户端安装用户，登录安装 Kafka 客户端的节点。

  b. 切换到 Kafka 客户端安装目录，例如“/opt/client”。

  cd /opt/client

  c. 执行以下命令，配置环境变量。

  source bigdata_env

  d. 执行以下命令，进行用户认证。（普通模式跳过此步骤）

  kinit *组件业务用户*

  e. 执行以下命令，切换到 Kafka 客户端安装目录。

  cd Kafka/kafka/bin

f. 执行以下命令，针对 Topic 修改配置和删除配置。

kafka-topics.sh --alter --topic <topic_name> --zookeeper <zookeeper_host:port>/kafka --config <name=value>

kafka-topics.sh --alter --topic <topic_name> --zookeeper <zookeeper_host:port>/kafka --delete-config <name>

例如：

kafka-topics.sh --alter --topic test1 --zookeeper 192.168.100.100:2181/kafka --config retention.ms=86400000

kafka-topics.sh --alter --topic test1 --zookeeper 192.168.100.100:2181/kafka --delete-config retention.ms

g. 执行以下命令，查询 topic 信息。

kafka-topics.sh --describe -topic <topic_name> --zookeeper <zookeeper_host:port>/kafka

13.26 如何读取“__consumer_offsets”内部 topic 的内容

用户问题

kafka 如何将 consumer 消费的 offset 保存在内部 topic“ __consumer_offsets”中？

处理步骤

步骤 1 以客户端安装用户，登录安装 Kafka 客户端的节点。

步骤 2 切换到 Kafka 客户端安装目录，例如“/opt/client”。

cd /opt/client

步骤 3 执行以下命令，配置环境变量。

source bigdata_env

步骤 4 执行以下命令，进行用户认证。（普通集群跳过此步骤）

kinit *组件业务用户*

步骤 5 执行以下命令，切换到 Kafka 客户端安装目录。

cd Kafka/kafka/bin

步骤 6 执行以下命令，获取 consumer offset metric 信息。

kafka-console-consumer.sh --topic __consumer_offsets --zookeeper <zk_host:port>/kafka --formatter "kafka.coordinator.group.GroupMetadataManager\$OffsetsMessageFormatter" --consumer.config <property file> --from-beginning

其中<property file>配置文件中需要增加如下内容。

```
exclude.internal.topics = false
```

例如：

kafka-console-consumer.sh --topic __consumer_offsets --zookeeper 10.5.144.2:2181/kafka --formatter "kafka.coordinator.group.GroupMetadataManager\$OffsetsMessageFormatter" --consumer.config ../config/consumer.properties --from-beginning

----结束

13.27 如何配置客户端 shell 命令的日志

用户问题

如何设置客户端 shell 命令的日志输出级别？

处理步骤

步骤 1 以客户端安装用户，登录安装 Kafka 客户端的节点。

步骤 2 切换到 Kafka 客户端安装目录，例如“/opt/client”。

cd /opt/client

步骤 3 切换到 Kafka 客户端配置目录。

cd Kafka/kafka/config

步骤 4 编辑 tools-log4j.properties 文件，将 WARN 修改为 INFO，并保存。

```
log4j.rootLogger=WARN, stderr

log4j.appender.stderr=org.apache.log4j.ConsoleAppender
log4j.appender.stderr.layout=org.apache.log4j.PatternLayout
log4j.appender.stderr.layout.ConversionPattern=[%d] %p %m (%c)%n
log4j.appender.stderr.Target=System.err
```

```
log4j.rootLogger=INFO, stderr

log4j.appender.stderr=org.apache.log4j.ConsoleAppender
log4j.appender.stderr.layout=org.apache.log4j.PatternLayout
log4j.appender.stderr.layout.ConversionPattern=[%d] %p %m (%c)%n
log4j.appender.stderr.Target=System.err
```

步骤 5 切换到 Kafka 客户端安装目录，例如“/opt/client”。

cd /opt/client

步骤 6 执行以下命令，配置环境变量。

source bigdata_env

步骤 7 执行以下命令，进行用户认证。（普通集群跳过此步骤）

kinit *组件业务用户*

步骤 8 执行以下命令，切换到 Kafka 客户端安装目录。

cd Kafka/kafka/bin

步骤 9 执行以下命令，获取 topic 信息，在控制台可见日志打印。

```
kafka-topics.sh --list --zookeeper 10.5.144.2:2181/kafka
[2017-02-17 14:34:27,005] INFO JAAS File name:
/opt/client/Kafka/./kafka/config/jaas.conf (org.I0Itec.zkclient.ZkClient)
[2017-02-17 14:34:27,007] INFO Starting ZkClient event thread.
(org.I0Itec.zkclient.ZkEventThread)
[2017-02-17 14:34:27,013] INFO Client environment:zookeeper.version=V100R002C10,
built on 05/12/2016 08:56 GMT (org.apache.zookeeper.ZooKeeper)
[2017-02-17 14:34:27,013] INFO Client environment:host.name=10-10-144-2
(org.apache.zookeeper.ZooKeeper)
[2017-02-17 14:34:27,013] INFO Client environment:java.version=1.8.0_72
(org.apache.zookeeper.ZooKeeper)
[2017-02-17 14:34:27,013] INFO Client environment:java.vendor=Oracle Corporation
(org.apache.zookeeper.ZooKeeper)
[2017-02-17 14:34:27,013] INFO Client environment:java.home=/opt/client/JDK/jdk/jre
(org.apache.zookeeper.ZooKeeper)
Test
__consumer_offsets
counter
test
test2
test3
test4
```

----结束

13.28 如何获取 Topic 的分布信息

用户问题

如何获取 Topic 在 Broker 实例的分布信息？

前置操作

- 前提条件

  已安装 Kafka、ZooKeeper 客户端。

- 操作步骤

  a. 以客户端安装用户，登录安装 Kafka 客户端的节点。

  b. 切换到 Kafka 客户端安装目录，例如“/opt/client”。

  cd /opt/client

c. 执行以下命令，配置环境变量。

source bigdata_env

d. 执行以下命令，进行用户认证。（普通集群跳过此步骤）

kinit *组件业务用户*

e. 执行以下命令，切换到 Kafka 客户端安装目录。

cd Kafka/kafka/bin

f. 执行 Kafka 相关命令，获取 Topic 分布信息和副本同步信息，观察返回结果。

kafka-topics.sh --describe --zookeeper <zk_host:port/chroot>

例如：

```
[root@mgtdat-sh-3-01-3 client]#kafka-topics.sh --describe --zookeeper
10.149.0.90:2181/kafka
Topic:topic1   PartitionCount:2  ReplicationFactor:2    Configs:
Topic: topic1  Partition: 0 Leader: 26 Replicas: 23,25 Isr: 26
Topic: topic1  Partition: 1 Leader: 24 Replicas: 24,23 Isr: 24,23
```

其中，Replicas 对应副本分布信息，Isr 对应副本同步信息。

处理方法 1

1. 在 ZooKeeper 中查询 Broker ID 的对应关系。

   sh zkCli.sh -server <zk_host:port>

2. 在 ZooKeeper 客户端执行如下命令。

   ls /kafka/brokers/ids

   get /kafka/brokers/ids/<查询出的 Broker id>

   例如：

```
[root@node-master1gAMQ kafka]# zkCli.sh -server node-master1gAMQ:2181
Connecting to node-master1gAMQ:2181
Welcome to ZooKeeper!
JLine support is enabled

WATCHER::

WatchedEvent state:SyncConnected type:None path:null
[zk: node-master1gAMQ:2181(CONNECTED) 0] ls /kafka/brokers/
ids     seqid   topics
[zk: node-master1gAMQ:2181(CONNECTED) 0] ls /kafka/brokers/ids
[1]
[zk: node-master1gAMQ:2181(CONNECTED) 1] get /kafka/brokers/ids/1
{"listener_security_protocol_map":{"PLAINTEXT":"PLAINTEXT","SSL":"SSL"},"endpoi
nts":["PLAINTEXT://192.168.2.242:9092","SSL://192.168.2.242:9093"],"rack":"/def
ault/rack0","jmx_port":21006,"host":"192.168.2.242","timestamp":"1580886124398"
,"port":9092,"version":4}
[zk: node-master1gAMQ:2181(CONNECTED) 2]
```

处理方法 2

获取节点和 broker ID 的对应关系

kafka-broker-info.sh --zookeeper <zk_host:port/chroot>

例如：

```
[root@node-master1gAMQ kafka]# bin/kafka-broker-info.sh --zookeeper
192.168.2.70:2181/kafka
Broker_ID    IP_Address
--------------------------
 1         192.168.2.242
```

13.29 Kafka 高可靠使用说明

Kafka 高可靠、高可用说明

Kafka 消息传输保障机制，可以通过配置不同的参数来保障消息传输，进而满足不同的性能和可靠性要求的应用场景。

- **Kafka 高可用、高性能**

  如果业务需要保证高可用和高性能，可以采用参数：

参数	默认值	说明
unclean.leader.election.enable	true	是否允许不在 ISR 中的副本被选举为 Leader，若设置为 true，可能会造成数据丢失。
auto.leader.rebalance.enable	true	是否使用 Leader 自动均衡功能。 如果设为 true，Controller 会周期性的为所有节点的每个分区均衡 Leader，将 Leader 分配给更优先的副本。
acks	1	需要 Leader 确认消息是否已经接收并认为已经处理完成。该参数会影响消息的可靠性和性能。 • acks=0 ：如果设置为 0，Producer 将不会等待服务端任何响应。消息将会被认为成功。 • acks=1 ：如果设置为 1，当副本所在 Leader 确认数据已写入，但是其不会等待所有的副本完全写入即返回响应。在这种情况下，如果 Leader 确认后但是副本未同步完成时 Leader 异常，那么数据就会丢失。 • acks=-1 ：如果设置为-1（all），意味着等待所有的同步副本确认后才认为成功，配合 min.insync.replicas 可以确保多副本写入成功，只要有一个副本保持活跃状态，记录将不会丢失。 说明 该参数在 kafka 客户端配置文件中配置。

参数	默认值	说明
min.insync.replicas	1	当 Producer 设置 acks 为-1 时，指定需要写入成功的副本的最小数目。

配置高可用、高性能的影响：

须知

配置高可用、高性能模式后，数据可靠性会降低。在磁盘故障、节点故障等场景下存在数据丢失风险。

- **Kafka 高可靠性配置说明**

  如果业务需要保证数据高可靠性，可以采用相关参数：

参数	建议值	说明
unclean.leader.election.enable	false	是否允许不在 ISR 中的副本被选举为 Leader。
acks	-1	Producer 需要 Leader 确认消息是否已经接收并认为已经处理完成。 acks=-1 需要表示等待在 ISR 列表的副本都确认接收到消息并处理完成才表示消息成功。配合 min.insync.replicas 可以确保多副本写入成功，只要有一个副本保持活跃状态，记录将不会丢失。 **说明** **该参数在 kafka 客户端配置文件中配置。**
min.insync.replicas	2	当 Producer 设置 acks 为-1 时，指定需要写入成功的副本的最小数目。 需要满足 min.insync.replicas <= replication.factor。

配置高可靠性的影响：

- 性能降低：

  需要所有的 ISR 列表副本，且满足最小成功的副本数确认写入成功。这样会导致单条消息时延增加，客户端处理能力下降，具体性能以现场实际测试数据为准。

- 可用性降低：

  不允许不在 ISR 中的副本被选举为 Leader。如果 Leader 下线时，其他副本均不在 ISR 列表中，那么该分区将保持不可用，直到 Leader 节点恢复。

需要所有的 ISR 列表副本，且满足最小成功的副本数确认写入成功。当分区的一个副本所在节点故障时，无法满足最小成功的副本数，那么将会导致业务写入失败。

配置影响

请根据业务场景对可靠性和性能要求进行评估，采用合理参数配置。

说明

- 对于价值数据，两种场景下建议 Kafka 数据目录磁盘配置 raid1 或者 raid5，从而提高单个磁盘故障情况下数据可靠性。
- 不同 Producer API 对应的 acks 参数名称不同
- 新 Producer API

  指 **org.apache.kafka.clients.producer.KafkaProducer** 中定义的接口，acks 配置为 acks。
- 旧 Producer API

  指 **kafka.producer.Producer** 中定义的接口，acks 配置名称为 request.required.acks。
- 参数配置项均为 Topic 级别可修改的参数，默认采用服务级配置。可针对不同 Topic 可靠性要求对 Topic 进行单独配置。

  例如，配置 Topic 名称为 test 的可靠性参数：

  kafka-topics.sh --zookeeper 192.168.1.205:2181/kafka --alter --topic test --config unclean.leader.election.enable=false --config min.insync.replicas=2

  其中 192.168.1.205 为 ZooKeeper 业务 IP 地址。
- 如果修改服务级配置需要重启 Kafka，建议在变更窗口做服务级配置修改。

13.30 Kafka 生产者写入单条记录过长问题

问题背景与现象

用户在开发一个 Kafka 应用，作为一个生产者调用新接口（org.apache.kafka.clients.producer.*）往 Kafka 写数据，单条记录大小为 1100055，超过了 kafka 配置文件 server.properties 中 message.max.bytes=1000012。用户修改了 Kafka 服务配置中 message.max.bytes 大小为 5242880，同时也将 replica.fetch.max.bytes 大小修改为 5242880 后，仍然无法成功。报异常大致如下：

```
..........
14749 [Thread-0] INFO  com.xxx.bigdata.kafka.example.NewProducer  - The
ExecutionException occured : {}.
java.util.concurrent.ExecutionException:
org.apache.kafka.common.errors.RecordTooLargeException: The message is 1100093
bytes when serialized which is larger than the maximum request size you have
configured with the max.request.size configuration.
at
org.apache.kafka.clients.producer.KafkaProducer$FutureFailure.<init>(KafkaProducer.
```

```
java:739)
at org.apache.kafka.clients.producer.KafkaProducer.doSend(KafkaProducer.java:483)
at org.apache.kafka.clients.producer.KafkaProducer.send(KafkaProducer.java:430)
at org.apache.kafka.clients.producer.KafkaProducer.send(KafkaProducer.java:353)
at com.xxx.bigdata.kafka.example.NewProducer.run(NewProducer.java:150)
Caused by: org.apache.kafka.common.errors.RecordTooLargeException: The message is
**** bytes when serialized which is larger than the maximum request size you have
configured with the max.request.size configuration.
.......
```

原因分析

经分析因为在写数据到 Kafka 时，Kafka 客户端会先比较配置项“max.request.size ”值和本次写入数据大小，若写入数据大小超过此配置项“max.request.size ”的缺省值，则抛出上述异常。

解决办法

步骤 1 在初始化 Kafka 生产者实例时，设置此配置项“max.request.size ”的值。

例如，参考本例，可以将此配置项设置为“5252880”：

```
      // 协议类型:当前支持配置为 SASL_PLAINTEXT 或者 PLAINTEXT
      props.put(securityProtocol, kafkaProc.getValues(securityProtocol,
"SASL_PLAINTEXT"));
      // 服务名
      props.put(saslKerberosServiceName, "kafka");
      props.put("max.request.size", "5252880");
      .......
```

----结束

13.31 Kakfa 消费者读取单条记录过长问题

问题背景与现象

和“Kafka 生产者写入单条记录过长问题”相对应的，在写入数据后，用户开发一个应用，以消费者调用新接口（org.apache.kafka.clients.consumer.*）到 Kafka 上读取数据，但读取失败，报异常大致如下：

```
..........
1687 [KafkaConsumerExample] INFO
org.apache.kafka.clients.consumer.internals.AbstractCoordinator  - Successfully
joined group DemoConsumer with generation 1
1688 [KafkaConsumerExample] INFO
org.apache.kafka.clients.consumer.internals.ConsumerCoordinator  - Setting newly
assigned partitions [default-0, default-1, default-2] for group DemoConsumer
2053 [KafkaConsumerExample] ERROR com.xxx.bigdata.kafka.example.NewConsumer  -
[KafkaConsumerExample], Error due to
org.apache.kafka.common.errors.RecordTooLargeException: There are some messages at
[Partition=Offset]: {default-0=177} whose size is larger than the fetch size
1048576 and hence cannot be ever returned. Increase the fetch size on the client
```

```
(using max.partition.fetch.bytes), or decrease the maximum message size the broker
will allow (using message.max.bytes).
2059 [KafkaConsumerExample] INFO  com.xxx.bigdata.kafka.example.NewConsumer  -
[KafkaConsumerExample], Stopped
.......
```

原因分析

经分析因为在读取数据时 Kafka 客户端会比较待读取数据大小和配置项“max.partition.fetch.bytes”值，若超过此配置项值，则抛出上述异常。

解决办法

步骤 1 在初始化建立 Kafka 消费者实例时，设置此配置项“max.partition.fetch.bytes”的值。

例如，参考本例，可以将此配置项设置为“5252880”：

```
......
// 安全协议类型
props.put(securityProtocol, kafkaProc.getValues(securityProtocol,
"SASL_PLAINTEXT"));
// 服务名
props.put(saslKerberosServiceName, "kafka");

props.put("max.partition.fetch.bytes","5252880");
......
```

----结束

13.32 Kafka 集群节点内多磁盘数据量占用高处理办法

用户问题

Kafka 流式集群节点内有多块磁盘的使用量很高。当达到 100%时就会造成 kafka 不可用如何处理？

问题现象

客户创建的 MRS Kafka 流式集群节点内有多块磁盘，由于分区不合理及业务原因导致某几个磁盘的使用量很高。当达到 100%时就会造成 kafka 不可用。

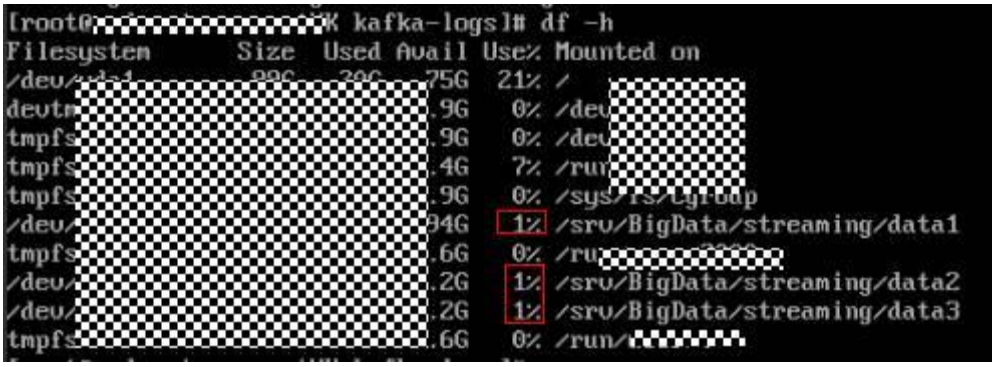
原因分析

需要提前干预处理磁盘数据，全局的 log.retention.hours 修改需要重启服务。为了不断服，可以将数据量大的单个 topic 老化时间根据需要改短。

处理步骤

步骤 1 登录 Kafka 集群的流式 Core 节点。

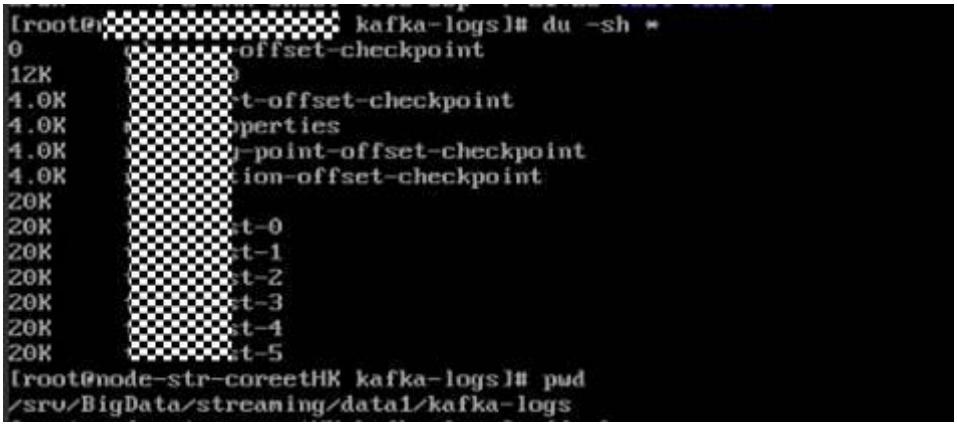
步骤 2　执行 **df　-h** 命令查看磁盘使用率。

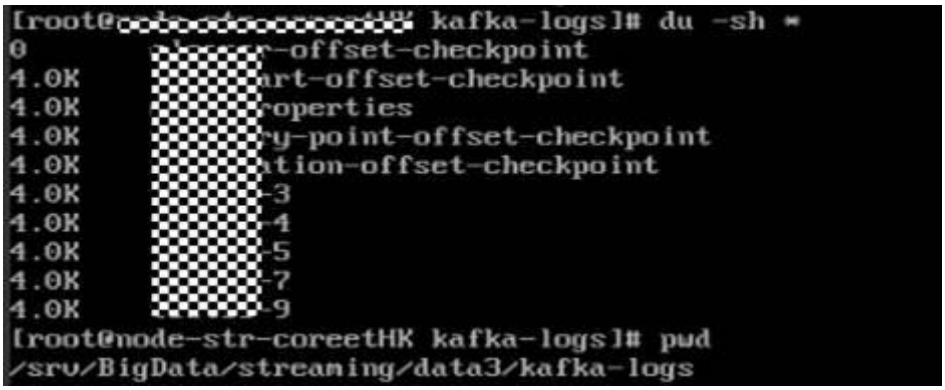


步骤 3　通过 kafka 配置文件 **opt/Bigdata/MRS_2.1.0/1_11_Broker/etc/server.properties** 中的配置项 **log.dirs** 获得数据存储目录。其中配置文件路径请根据时间环境的集群版本修改，当磁盘有多块时，该配置项有多个，逗号间隔。

步骤 4　使用 **cd** 命令进入使用率较高的磁盘对应的步骤 3 中获取的数据存储目录下。

步骤 5　使用 **du -sh *** 命令打印出当前 topic 的名称及大小。





步骤 6 由于 kafka 的全局的数据保留时间默认是 7 天。部分 topic 由于业务写入量大，而这些 topic 的分区正好在上面使用量高的磁盘上，因此导致磁盘使用率较高。

- 可以通过修改全局数据的保留期为较短时间来释放磁盘空间，该方式需要重启 Kafka 服务才能生效，可能会影响业务运行。具体请参见步骤 7。
- 可以单独将 topic 的数据保留期改为较短时间来释放磁盘空间，该方式无需重启 Kafka 服务即可生效。具体请参见步骤 8。

步骤 7 登录 Manager 页面，在 Kafka 的服务配置页面，切换为“全部配置”并搜索“log.retention.hours”配置项，该值默认为 7 天，请根据需要进行修改。

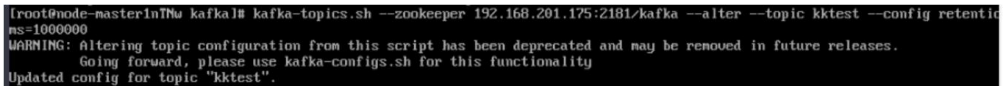
步骤 8 可以单独将这些磁盘上的 topic 的数据老化时间修改为较短时间来解决该问题。

1. 查看 topic 数据过期时间。

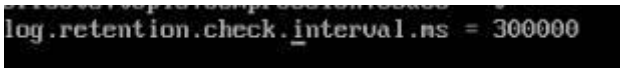
   bin/kafka-topics.sh --describe --zookeeper <*ZooKeeper 集群业务 IP*>:2181/kafka --topic kktest

2. 设置 topic 数据过期时间，其中**--topic** 表示具体 topic 名称，retention.ms=具体的数据过期时间，单位是毫秒。

   kafka-topics.sh --zookeeper <*ZooKeeper 集群业务 IP*>:2181/kafka --alter --topic kktest --config retention.ms=1000000



设置数据过期时间之后会可能不会立刻执行，删除操作在参数 **log.retention.check.interval.ms** 所规定时间之后开始执行删，可以通过查看 kafka 的 server.log 检索是否有 delete 字段有判断删除操作是否生效，有 delete 字段则表示已经生效，也可以通过执行 **df -h** 命令查看磁盘的数据量占用情况判断设置是否生效。



----结束

14 使用 Oozie

14.1 当并发提交大量 oozie 任务时，任务一直没有运行

用户问题

并发提交大量 oozie 任务的时候，任务一直没有运行。

问题现象

并发提交大量 oozie 任务的时候，任务一直没有运行。

原因分析

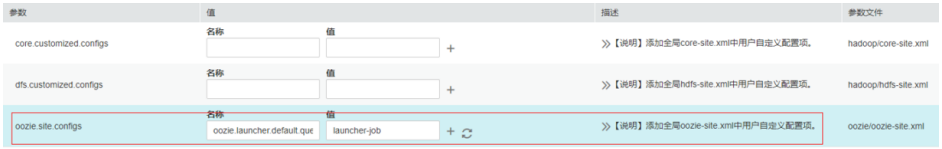
Oozie 提交任务会先启动一个 oozie-launcher，然后由 oozie-launcher 提交真正的作业运行。默认情况下 launcher 和真实作业会在同一个队列中。

当并发提交大量 oozie 任务的时候就有可能出现启动了一堆 oozie-launcher，将队列的资源耗完，而没有更多资源启动真实作业，最终导致任务一直没有运行。

处理步骤

步骤 1 参考“用户指南 > 管理现有集群 > 租户管理 > 添加租户”章节新建一个队列给 oozie 使用，也可以直接使用创建 MRS 集群时生成的 launcher-job 队列。

步骤 2 在 Manager 页面选择“集群 > 服务 > Oozie > 配置”，搜索参数“oozie.site.configs”，在值列添加名称“oozie.launcher.default.queue”，值为“launcher-job”。



----结束

14.2 Oozie 调度 HiveSQL 作业报错处理

问题现象

MRS 3.x 集群 Oozie 调度 Hive 作业任务失败，查看日志 HiveSQL，实际上是运行成功的，但 Yarn 任务总体失败，报错如下：

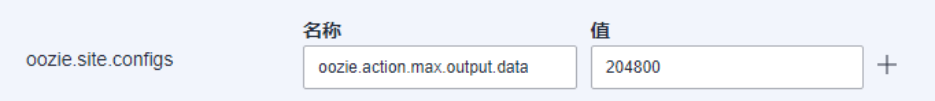
```
java.io.ioException:output.properties data exceeds its limit [2048]
```

原因分析

由于 HiveSQL 脚本中一次提交的作业量太大，其中包含的信息超过 Oozie Launcher 一次容许的最大值 2KB（2048 Bytes），需要调大默认的输出参数值，调整该参数不会对集群性能造成影响。

处理步骤

步骤 1 登录 Manager 页面，选择“集群 > 服务 > Oozie > 配置 > 全部配置 > 自定义”，在“oozie-site.xml”的配置项中添加参数“oozie.action.max.output.data”，值为“204800”，如下所示：



步骤 2 添加完后，保存并重启 Oozie 服务。

步骤 3 重新执行 Oozie 调度作业，查看结果，已正常。

----结束

15 使用 Presto

15.1 配置 sql-standard-with-group 创建 schema 失败报 Access Denied

用户问题

配置 sql-standard-with-group 创建 schema 失败，报 Access Denied 的错误，如何处理？

问题现象

```
CREATE SCHEMA hive.sf2 WITH (location = 'obs://obs-zy1234/sf2');Query
20200224_031203_00002_g6gzy failed: Access Denied: Cannot create schema sf2
```

原因分析

presto 创建 schema 需要 hive 的管理者权限。

处理步骤

MRS Manager 界面操作：

- 方法一：
  a. 登录 MRS Manager 页面，选择“系统设置 > 用户管理”。
  b. 在对应用户所在行的“操作”列，单击“修改”。
  c. 单击“选择并绑定角色”，为用户添加 System_administrator 的权限。
  d. 单击“确定”完成修改。
- 方法二：
  a. 登录 MRS Manager 页面，选择“系统设置 > 角色管理”。
  b. 单击“添加角色”，并配置如下参数。
     - 角色名称：配置角色名称，例如 hive_admin。
     - 权限：选择“Hive”，并勾选 Hive Admin Privilege。
  c. 单击“确定”保存角色。

d. 选择“系统设置 > 用户管理”。
e. 在对应用户所在行的“操作”列，单击“修改”。
f. 单击“选择并绑定角色”，为用户添加新创建的 hive_admin 的权限。
g. 单击“确定”完成修改。

FusionInsight Manager 界面操作：

- 方法一：
  a. 登录 FusionInsight Manager 页面，选择“系统 > 权限 > 用户”。
  b. 在对应用户所在行的“操作”列，单击“修改”。
  c. 单击角色后的“添加”，为用户添加 System_administrator 的权限。
  d. 单击“确定”完成修改。
- 方法二：
  a. 登录 FusionInsight Manager 页面，选择“系统 > 权限 > 角色”。
  b. 单击“添加角色”，并配置如下参数。
     - 角色名称：配置角色名称，例如 hive_admin。
     - 配置资源权限：选择“Hive”，并勾选“Hive 管理员权限”。
  c. 单击“确定”保存角色。
  d. 选择“系统 > 权限 > 用户”。
  e. 在对应用户所在行的“操作”列，单击“修改”。
  f. 单击角色后的“添加”，为用户添加新创建的 hive_admin 的权限。
  g. 单击“确定”完成修改。

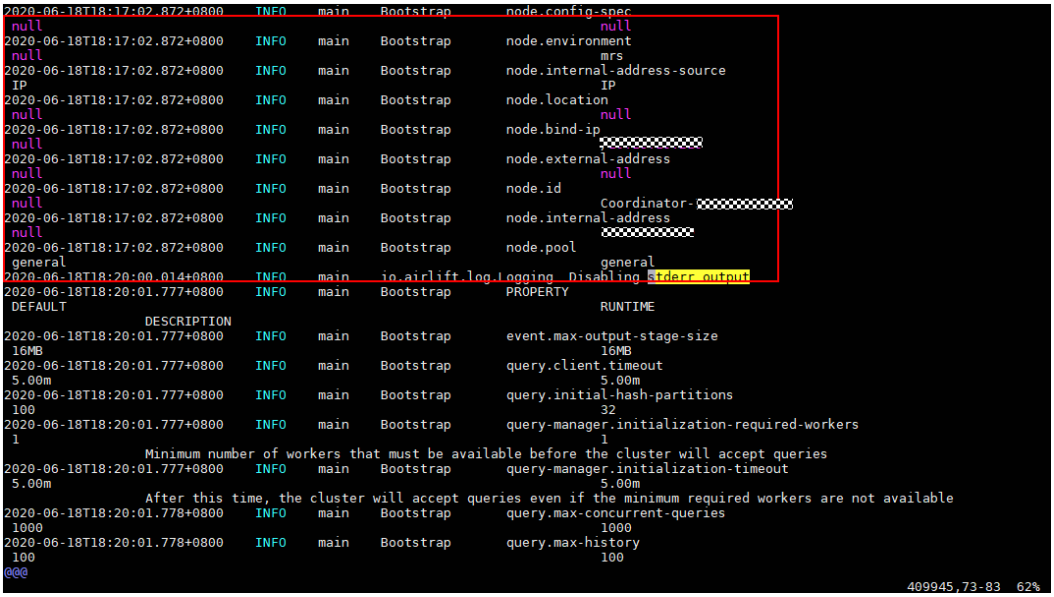
15.2 Presto 的 coordinator 无法正常启动

用户问题

Presto 的 coordinator 未知原因被 kill，或者 presto 的 coordinator 进程无法正常启动。

问题现象

Presto 的 coordinator 无法正常启动，Manager 页面上显示 presto coordinator 进程正常启动且状态正常，但查看后台日志 coordinator 进程未真正启动，只有如下日志：



presto 的 coordinator 未真正启动即被 Kill 了，不再打印其他日志，查看 presto 的其他日志也未发现为何被 kill。

原因分析

presto 的健康检查脚本的端口检查逻辑中未做好端口的区分。

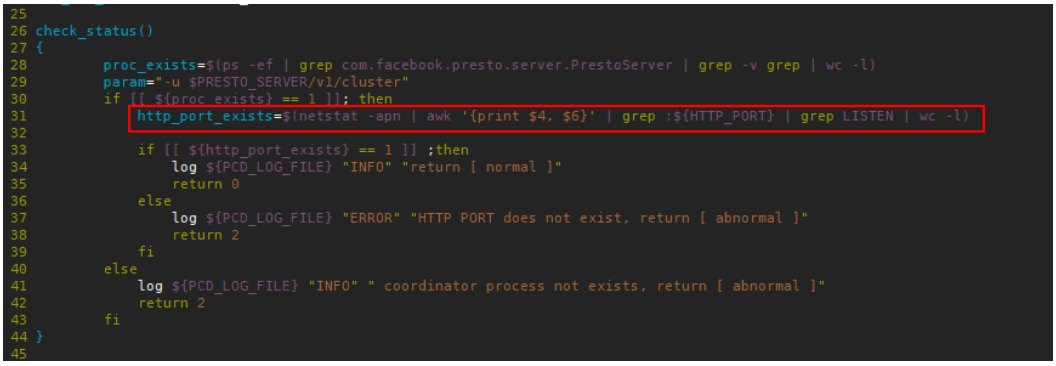
处理步骤

步骤 1 使用工具分别登录集群的 Master 节点执行如下操作。

步骤 2 执行如下命令编辑文件。

vim /opt/Bigdata/MRS_xxx/install/FusionInsight-Presto-*/ha/module/harm/plugin/script/pcd.sh

该文件中的第 31 行修改为“http_port_exists=$(netstat -apn | awk '{print $4, $6}' | grep :${HTTP_PORT} | grep LISTEN | wc -l)”。



步骤 3 保存如上修改，再在 manager 页面上选择“服务管理 > Presto > 实例”，重启 Coordinator 进程。

----结束

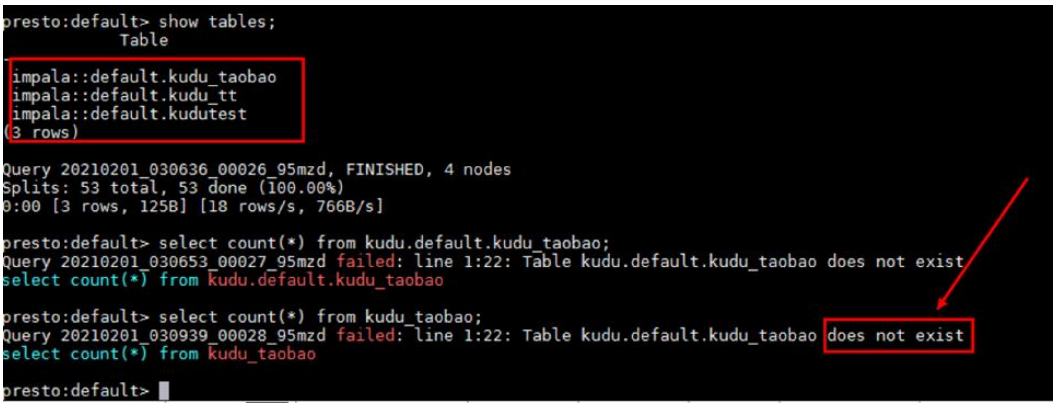
15.3 Presto 查询 Kudu 表报错

用户问题

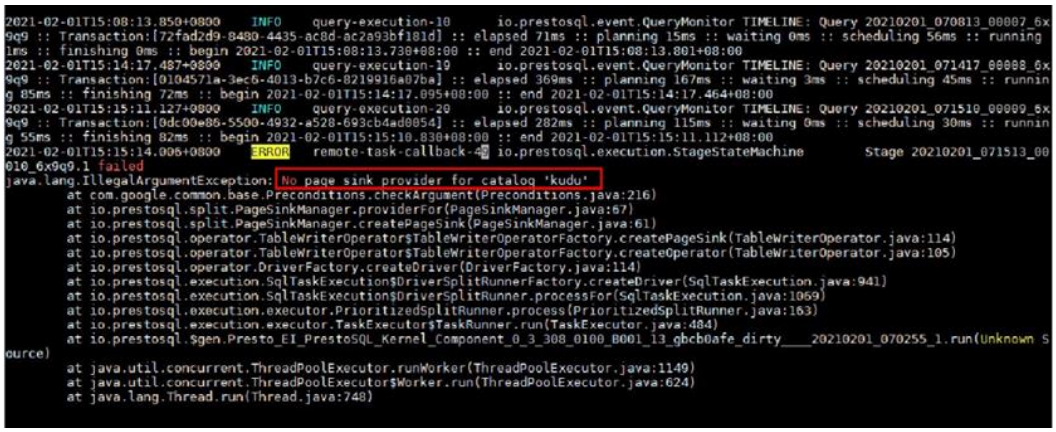
使用 presto 查询 Kudu 表报错。

问题现象

使用 presto 查询 Kudu 表，报表找不到的错误：



后台报错：



原因分析

在实际的运行节点（worker 实例所在节点）没有 kudu 相关配置。

处理步骤

步骤 1 在集群 presto 所有的 worker 实例节点添加配置文件 kudu.properties。

配置文件保存路径：/opt/Bigdata/MRS_xxx/1_x_Worker/etc/catalog/ (请根据集群实际版本修改路径)

配置文件内容：

```
connector.name=kudu
kudu.client.master-
addresses=KuduMasterIP1:port,KuduMasterIP2:port,KuduMasterIP3:port
```

说明

- KuduMaster 节点 IP 和端口请根据实际情况填写。
- 为配置文件添加和文件保存路径下其他文件一致的文件权限、属组。

步骤 2 修改完成之后，请在集群详情页面选择“组件管理 > Kudu”，单击“更多 > 重启服务”。

----结束

15.4 Presto 查询 Hive 表无数据

用户问题

使用 presto 查询 Hive 表无数据。

问题现象

通过 Tez 引擎执行 union 相关语句写入的数据，Presto 无法查询。

原因分析

由于 Hive 使用 Tez 引擎在执行 union 语句时，生成的输出文件会保存在 HIVE_UNION_SUBDIR 目录中，而 Presto 默认不读取子目录下的文件，所以没有读取到 HIVE_UNION_SUBDIR 目录下的数据。

处理步骤

步骤 1 在集群详情页面选择“组件管理 > Presto > 服务配置”。

步骤 2 切换“基础配置”为全部配置“。

步骤 3 在左侧导航处选择“Presto > Hive”，在 catalog/hive.properties 文件中增加 hive.recursive-directories 参数，值为 true。

步骤 4 单击“保存配置”并勾选“重新启动受影响的服务或实例。”。

----结束

15.5 MRS presto 查询报错

用户问题

客户报障 presto 查询语句报错：The node may have　crashed or be under too much load，具体如下图所示：

图15-1 报错信息

问题分析

1. 查询到客户集群 id，先要到运维授权，登录客户 master 节点，进入/var/log/Bigdata 下找 presto 的日志进行查找。

   其中报错和客户报障界面相同。

2. 根据报错信息在到 192.168.0.243 节点去查看 presto worker 实例的进程日志。日志显示：

```
java.lang.OutOfMemoryError: Java heap space
```

3. 根据报错预判发生了 oom 导致查询报错。

处理步骤

步骤 1　在 manager 页面查找 work 的 jvm 参数（xmx）。

图15-2 jvm 参数配置

步骤 2　将 xmx 后面的 1024 改成 2048，然后保存配置。

步骤 3　重启 presto，再进行查询，问题解决不在报错。

----结束

15.6 MRS 集群如何使用公网访问 Presto

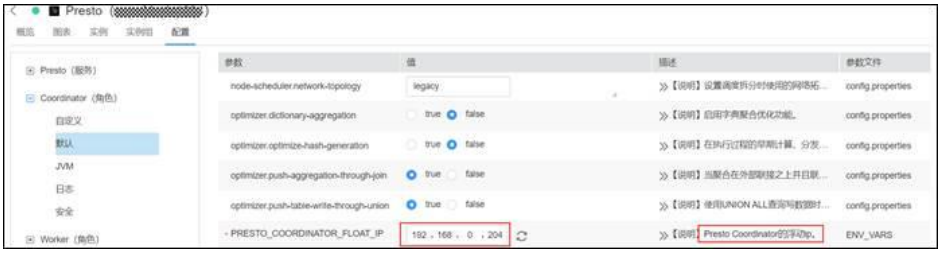
用户问题

客户调测 Presto jdbc 样例代码，需要使用公网访问。

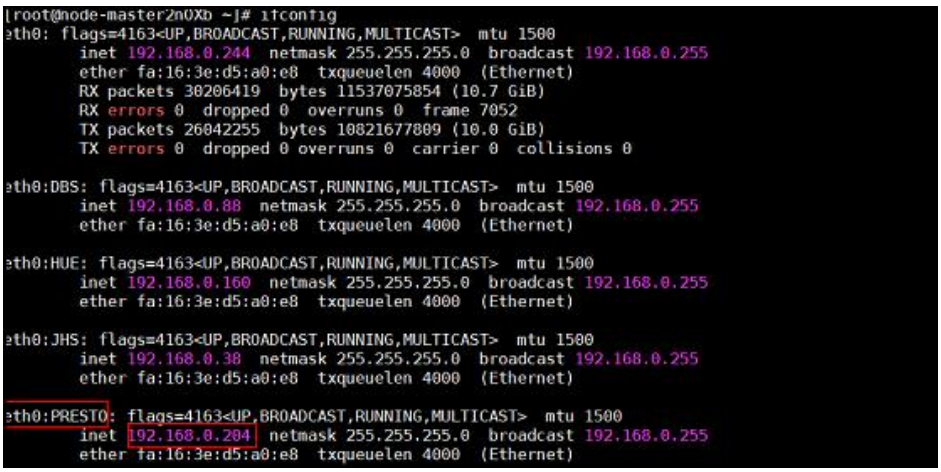
问题分析

登录 MRS manager 上查看 Presto 全部配置，Coordinator 角色配置：

PRESTO_COORDINATOR_FLOAT_IP = 私网 ip 地址



登录 Coordinator 角色所在主实例节点确认该地址为网卡：eth0:PRESTO。



Presto 服务端口：非安全集群为 7520；安全集群为 7521；

因此公网访问需要给对应的浮动网卡绑定公网地址，然后填写正确的 JDBC URL 即可

URL 格式：

jdbc:presto://example_ip:7520/Catalog/schema

jdbc:presto://example_ip:7521/Catalog/schema

处理步骤

步骤 1 在 MRS manager 管理页面找到 presto 组件，记录 presto 的内网浮动 ip 地址；

步骤 2 在 VPC 上创建一个弹性公网 IP 地址，并在 VPC 控制台上找到 MRS 集群的子网，然后找到 Presto 的浮动 IP，并给改浮动 IP 绑上弹性公网 IP 地址；

步骤 3　在 MRS 服务集群的安全组中放通源地址到 MRS 服务 presto 的端口访问，然后测试；（以非安全集群为例，catalog 为 hive）

```
[root@node-masterlmnma ~]# presto_cli.sh --server http://119.3.164.55:7520 --catalog hive --schema bigdata
--server http://119.3.164.55:7520 --catalog hive --schema bigdata
presto:bigdata> select * from eimrs;
 id | name | worker | happies | age
----+------+--------+---------+-----
 0  |      |        |         |  30
 1  |      |        |         |  28
 2  |      |        |         |  32
(3 rows)

Query 20210731_050300_00007_p4y4b, FINISHED, 1 node
Splits: 19 total, 19 done (100.00%)
0:00 [3 rows, 96B] [10 rows/s, 326B/s]

presto:bigdata>
```

----结束

16 使用 Spark

16.1 Spark 应用下修改 split 值时报错

用户问题

在 Spark 应用下修改 split 值时报错。

问题现象

客户需要通过修改一个 split 最大值来实现多个 mapper ，从而达到提速的目的， 但是目前执行 **set $参数**命令修改 Hive 的配置时报错。

```
0: jdbc:hive2://192.168.1.18:21066/> set mapred.max.split.size=1000000;
Error: Error while processing statement: Cannot nodify mapred.max.split.size at
runtime. It is not in list of params that are allowed to be modified at
runtime( state=42000,code=1)
```

原因分析

- 在安全模式下配置白名单启停参数 **hive.security.whitelist.switch** 时，需要运行的参数必须在 **hive.security.authorization.sqlstd.confwhitelist** 中配置。
- 默认白名单中没有包含 **mapred.max.split.size** 参数，所以运行的时候会提示不允许。

处理步骤

步骤 1 搜索 **hive.security.authorization.sqlstd.confwhitelist.append**，把 **mapred.max.split.size** 加进 **hive.security.authorization.sqlstd.confwhitelist.append** 中，详细信息可参考。

步骤 2 修改完成后，保存配置，重启 Hive 组件。

步骤 3 执行 **set mapred.max.split.size=1000000;**，系统不再报错，则表示修改成功。

----结束

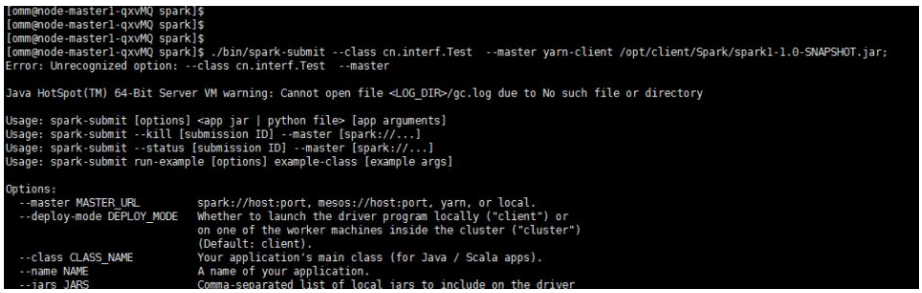
16.2 使用 Spark 时报错

用户问题

在使用 spark 时，集群运行失败。

问题现象

客户在使用 spark 组件时，集群运行失败。



原因分析

- 执行命令时，引入非法字符
- 上传的 jar 包属主属组有问题

处理步骤

步骤 1 检查用户命令**./bin/spark-submit --class cn.interf.Test --master yarn-client** *客户端安装目录***/Spark/spark1-1.0-SNAPSHOT.jar;**，排查是否引入非法字符。

步骤 2 如果是，修改非法字符，重新执行命令。

步骤 3 重新执行命令后，发生其他错误，查看该 jar 包的属主属组，发现全为 root。

步骤 4 修改 jar 包的属主属组为 omm:wheel，重新执行成功。

----结束

16.3 引入 jar 包不正确，导致 Spark 任务无法运行

用户问题

执行 Spark 任务，任务无法运行。

问题现象

执行 Spark 任务，任务无法运行。

原因分析

执行 Spark 任务时，引入的 jar 包不正确，导致 Spark 任务运行失败。

处理步骤

步骤 1 登录任意 Master 节点。

步骤 2 执行 **cd /opt/Bigdata/MRS_*/install/FusionInsight-Spark-*/spark/examples/jars** 命令，查看样例程序的 jar 包。

说明

jar 包名最多为 1023 字符，不能包含;|&>,<'$特殊字符，且不可为空或全空格。

步骤 3 检查 OBS 桶上的执行程序，执行程序可存储于 HDFS 或者 OBS 中，不同的文件系统对应的路径存在差异。

说明

- OBS 存储路径：以“obs://”开头。示例：obs://wordcount/program/hadoop-mapreduce-examples-2.7.x.jar
- HDFS 存储路径：以“/user”开头。Spark Script 需要以“.sql”结尾，MR 和 Spark 需要以“.jar”结尾。sql、jar 不区分大小写。

----结束

16.4 Spark 任务由于内存不够或提交作业时未添加 Jar 包，作业卡住

用户问题

Spark 提交作业内存不足或提交作业时未添加 Jar 包导致任务长时间处于 pending 状态或者运行中内存溢出。

问题现象

使用 Spark 提交作业后，长期卡住不动。反复运行作业后报错，内容如下：

```
Exception in thread "main" org.apache.spark.SparkException: Job aborted due to
stage failure:
Aborting TaskSet 3.0 because task 0 (partition 0) cannot run anywhere due to node
and executor blacklist.
Blacklisting behavior can be configured via spark.blacklist.*.
```

原因分析

内存不足或提交作业时未添加 Jar 包，导致 Spark 提交的作业任务长时间处于 pending 状态。

处理步骤

步骤 1 检查提交作业时是否添加 Jar 包。

- 是，执行步骤 2。
- 否，添加 Jar 包，执行作业正常，**操作结束**。如果执行作业任务长时间处于 pending 状态，执行步骤 2。

步骤 2 登录 MRS Console 页面，在现有集群中，选择集群名称，在“节点信息”页面，查看当前集群的节点规格。

步骤 3 提高 nodemanager 进程所持有的集群资源。

MRS Manager 界面操作：

1. 登录 MRS Manager 页面，选择“服务管理 > Yarn > 服务配置”。
2. 在“参数类别”中选择“全部配置”，然后在搜索框中搜索 **yarn.nodemanager.resource.memory-mb**，查看该参数值。建议配置成节点物理内存总量的 75%-90%。

FusionInsight Manager 界面操作：

1. 登录 FusionInsight Manager。选择“集群 > 服务 > Yarn”。
2. 单击“配置”，选择“全部配置”。然后在搜索框中搜索 **yarn.nodemanager.resource.memory-mb**，查看该参数值。建议配置成节点物理内存总量的 75%-90%。

步骤 4 修改 Spark 的服务配置。

MRS Manager 界面操作：

1. 登录 MRS Manager 页面，选择“服务管理”>“Spark” >“服务配置”。
2. 在“参数类别”中选择“全部配置”，然后在搜索框中搜索 **spark.driver.memory** 和 **spark.executor.memory**

   根据作业的需要调大或者调小该值，具体以提交的 Spark 作业的复杂度和内存需要为参考（一般调大）。

FusionInsight Manager 界面操作：

1. 登录 FusionInsight Manager。选择“集群 > 服务 > Spark”。
2. 单击“配置”，选择“全部配置”。然后在搜索框中搜索 **spark.driver.memory** 和 **spark.executor.memory**，根据作业的需要调大或者调小该值，具体以提交的 Spark 作业的复杂度和内存需要为参考（一般调大）。

说明

- 如果使用到 SparkJDBC 作业，搜索并修改 **SPARK_EXECUTOR_MEMORY** 和 **SPARK_DRIVER_MEMORY** 两个参数取值，具体以提交的 Spark 作业的复杂度和内存需要为参考（一般调大）。
- 如果对核数有要求，可以搜索并修改 **spark.driver.cores** 和 **spark.executor.cores** 的核数取值。

步骤 5 Spark 依赖内存做计算，如果以上还是不能满足任务的提交需要，建议扩容集群。

----结束

16.5 运行 Spark 报错

用户问题

运行 Spark 作业报找不到指定的类。

问题现象

运行 Spark 作业报找不到指定的类。报错内容如下：

```
Exception encountered |
org.apache.spark.internal.Logging$class.logError(Logging.scala:91)
org.apache.hadoop.hbase.DoNotRetryIOException: java.lang.ClassNotFoundException:
org.apache.phoenix.filter.SingleCQKeyValueComparisonFilter
```

原因分析

用户配置的默认路径不正确。

处理步骤

步骤 1 登录任意 Master 节点。

步骤 2 修改 Spark 客户端目录下的配置文件。

执行 **vim** *客户端安装目录*/**Spark/spark/conf/spark-defaults.conf** 命令，打开 spark-defaults.conf 文件，设置“spark.executor.extraClassPath” 取值为“${PWD}/*”。

----结束

16.6 Driver 端提示 executor memory 超限

问题背景与现象

内存超限导致提交 Spark 任务失败。

原因分析

在 Driver 日志中直接打印申请的 executor memory 超过集群限制。

```
16/02/06 14:11:25 INFO Client: Verifying our application has not requested more
than the maximum memory capability of the cluster (6144 MB per container)
16/02/06 14:11:29 ERROR SparkContext: Error initializing SparkContext.
java.lang.IllegalArgumentException: Required executor memory (10240+1024 MB) is
above the max threshold (6144 MB) of this cluster!
```

Spark 任务提交至 Yarn 上面，运行 task 的 executor 使用的资源受 yarn 的管理。从报错信息可看出，用户申请启动 executor 时，指定 10G 的内存，超出了 Yarn 设置的每个 container 的最大内存的限制，导致任务无法启动。

解决办法

修改 Yarn 的配置，提高对 container 的限制。如可通过调整“yarn.scheduler.maximum-allocation-mb”参数的大小，可控制启动的 executor 的资源，修改之后要重启 Yarn 服务。

配置修改方法：

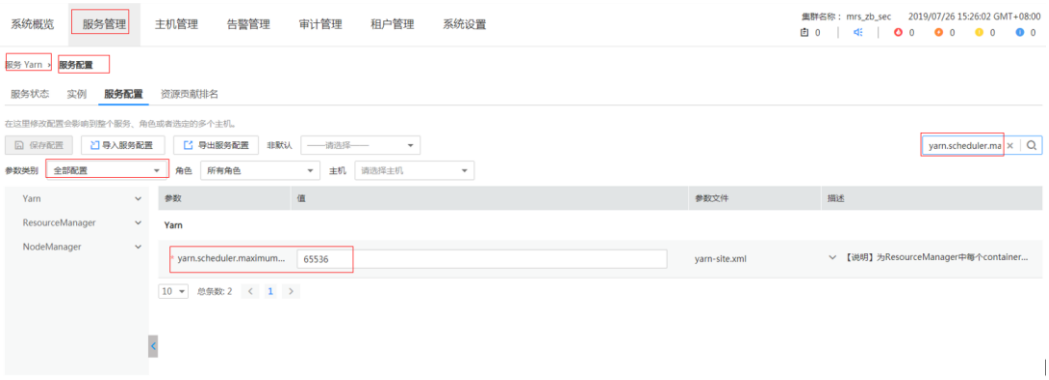
MRS Manager 界面操作：

步骤 1 登录 MRS Manager 页面。

步骤 2 选择“服务管理 > Yarn > 服务配置”将“参数类别”修改为“全部配置”。

步骤 3 在“搜索”栏输入“yarn.scheduler.maximum-allocation-mb”修改参数并保存重启服务。如下图所示：

图16-1 修改 Yarn 服务参数



----结束

FusionInsight Manager 界面操作：

步骤 1 登录 FusionInsight Manager 页面。

步骤 2 选择“集群 > 服务 > Yarn”，单击“配置”，选择“全部配置”。

步骤 3 在“搜索”栏输入“yarn.scheduler.maximum-allocation-mb”修改参数并保存重启服务。

----结束

16.7 Yarn-cluster 模式下，Can't get the Kerberos realm 异常

问题背景与现象

认证异常导致提交 Spark 任务失败。

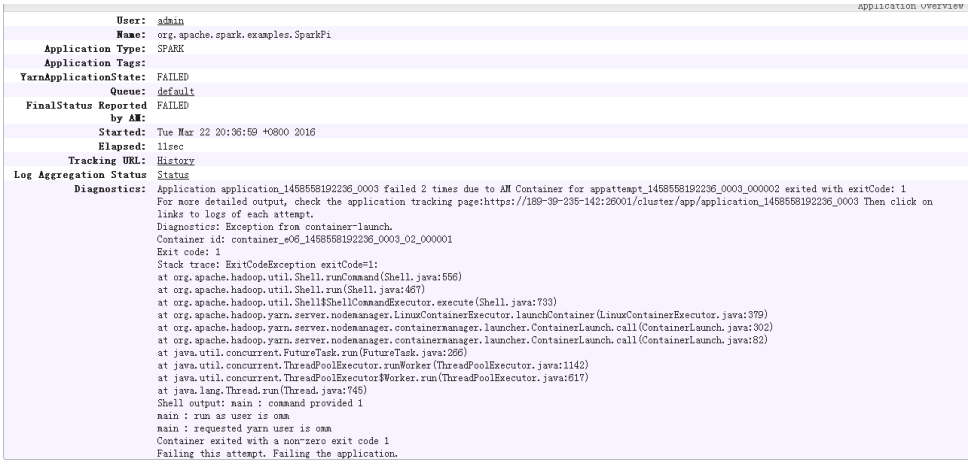
原因分析

1. 在 driver 端打印异常找不到连接 hdfs 的 token，报错如下：

```
16/03/22 20:37:10 WARN Client: Exception encountered while connecting to the
server :
org.apache.hadoop.ipc.RemoteException(org.apache.hadoop.security.token.SecretMa
nager$InvalidToken): token (HDFS_DELEGATION_TOKEN token 192 for admin) can't be
found in cache
16/03/22 20:37:10 WARN Client: Failed to cleanup staging
dir .sparkStaging/application_1458558192236_0003
org.apache.hadoop.ipc.RemoteException(org.apache.hadoop.security.token.SecretMa
nager$InvalidToken): token (HDFS_DELEGATION_TOKEN token 192 for admin) can't be
found in cache
```

2. 在 Yarn 原生页面显示 am 启动两次均失败，任务退出，如图 16-2 信息：

图16-2 am 启动失败



3. 查看 ApplicationMaster 日志看到如下异常信息：

```
Exception in thread "main" java.lang.ExceptionInInitializerError
Caused by: org.apache.spark.SparkException: Unable to load YARN support
Caused by: java.lang.IllegalArgumentException: Can't get Kerberos realm
Caused by: java.lang.reflect.InvocationTargetException
Caused by: KrbException: Cannot locate default realm
Caused by: KrbException: Generic error (description in e-text) (60) - Unable to
locate Kerberos realm
org.apache.hadoop.hive.metastore.MetaStoreUtils.newInstance(MetaStoreUtils.java
:1410)
... 86 more
Caused by: javax.jdo.JDOFatalInternalException: Unexpected exception caught.
```

```
NestedThrowables:java.lang.reflect.InvocationTargetException
... 110 more
```

4. 执行**./spark-submit --class yourclassname --master yarn-cluster /yourdependencyjars** 任务以 yarn-cluster 模式提交任务，driver 端会在集群中启用，由于加载的是客户端的 spark.driver.extraJavaOptions，在集群节点上对应路径下找不到对应的 kdc.conf 文件，无法获取 kerberos 认证所需信息，导致 am 启动失败。

解决办法

在客户端提交任务时，在命令行中配置自定义的 spark.driver.extraJavaOptions 参数这样任务运行时就不会自动加载客户端路径下 spark-defaults.conf 中的 spark.driver.extraJavaOptions；或者在启动 spark 任务时，通过--conf 来指定 driver 的配置，如下（此处 spark.driver.extraJavaOptions“=”号后面的引号部分不能缺少）。

./spark-submit -class yourclassname --master yarn-cluster --conf spark.driver.extraJavaOptions="

-Dlog4j.configuration=file:/opt/client/Spark/spark/conf/log4j.properties -Djetty.version=x.y.z -Dzookeeper.server.principal=zookeeper/hadoop.794bbab6_9505_44cc_8515_b4eddc84e6c1.com -Djava.security.krb5.conf=/opt/client/KrbClient/kerberos/var/krb5kdc/krb5.conf -Djava.security.auth.login.config=/opt/client/Spark/spark/conf/jaas.conf -Dorg.xerial.snappy.tempdir=/opt/client/Spark/tmp -Dcarbon.properties.filepath=/opt/client/Spark/spark/conf/carbon.properties" ../yourdependencyjars

16.8 JDK 版本不匹配启动 spark-sql，spark-shell 失败

问题背景与现象

JDK 版本不匹配导致客户端启动 spark-sql，spark-shell 失败。

原因分析

1. 在 Driver 端打印异常如下：

```
 Exception Occurs: BadPadding 16/02/22 14:25:38 ERROR Schema: Failed
initialising database. Unable to open a test connection to the given database.
JDBC url = jdbc:postgresql://ip:port/sparkhivemeta, username = spark.
Terminating connection pool (set lazyInit to true if you expect to start your
database after your app).
```

2. Sparksql 任务使用时，需要访问 DBService 以获取元数据信息，在客户端需要解密密文来访问，在使用过程中，用户没有按照流程操作，没有执行配置环境变量操作，且在其客户端环境变量中存在默认的 jdk 版本，导致在执行解密过程中调用的解密程序执行解密异常，会引起用户被锁。

解决办法

步骤 1 使用 **which java** 命令查看默认的 java 命令是否是客户端的 java。

步骤 2 如果不是，请按正常的客户端执行流程。

source ${client_path}/bigdata_env

kinit *用户名*，然后输入用户名对应的密码，启动任务即可。

----结束

16.9 Yarn-client 模式提交 ApplicationMaster 尝试启动两次失败

问题背景与现象

Yarn-client 模式提交任务 AppMaster 尝试启动两次失败。

原因分析

1. Driver 端异常:

```
16/05/11 18:10:56 INFO Client:
client token: N/A
diagnostics: Application application_1462441251516_0024 failed 2 times due to
AM Container for appattempt_1462441251516_0024_000002 exited with  exitCode: 10
For more detailed output, check the application tracking
page:https://hdnode5:26001/cluster/app/application_1462441251516_0024 Then
click on links to logs of each attempt.
Diagnostics: Exception from container-launch.
Container id: container_1462441251516_0024_02_000001
```

2. 在 ApplicationMaster 日志中，异常如下:

```
2016-05-12 10:21:23,715 | ERROR | [main] | Failed to connect to driver at
192.168.30.57:23867, retrying ... |
org.apache.spark.Logging$class.logError(Logging.scala:75)
2016-05-12 10:21:24,817 | ERROR | [main] | Failed to connect to driver at
192.168.30.57:23867, retrying ... |
org.apache.spark.Logging$class.logError(Logging.scala:75)
2016-05-12 10:21:24,918 | ERROR | [main] | Uncaught exception:  |
org.apache.spark.Logging$class.logError(Logging.scala:96)
org.apache.spark.SparkException: Failed to connect to driver!
at
org.apache.spark.deploy.yarn.ApplicationMaster.waitForSparkDriver(ApplicationMa
ster.scala:426)
at
org.apache.spark.deploy.yarn.ApplicationMaster.runExecutorLauncher(ApplicationM
aster.scala:292)
…
2016-05-12 10:21:24,925 | INFO  | [Thread-1] | Unregistering ApplicationMaster
with FAILED (diag message: Uncaught exception: org.apache.spark.SparkException:
Failed to connect to driver!) |
org.apache.spark.Logging$class.logInfo(Logging.scala:59)
```

Spark-client 模式任务 Driver 运行在客户端节点上(通常是集群外的某个节点)，启动时先在集群中启动 AppMaster 进程，进程启动后要向 Driver 进程注册信息，注册成功后，任务才能继续。从 AppMaster 日志中可以看出，无法连接至 Driver，所以任务失败。

解决办法

步骤 1 请检查 Driver 进程所在的 IP 是否可以 **ping** 通。

步骤 2 启动一个 sparkpi 任务，在 console 端会有类似如下打印信息。

```
16/05/11 18:07:20 INFO Remoting: Remoting started; listening on
addresses :[akka.tcp://sparkDriver@192.168.1.100:23662]
16/05/11 18:07:20 INFO Utils: Successfully started service 'sparkDriver' on port
23662.
```

步骤 3 在该节点，也就是步骤 2 中示例的 192.168.1.100 上执行 **netstat - anp | grep 23662** 看下此端口是否打开，如下打印标明，相关端口是打开的。

```
tcp       0      0  ip:port    :::*                    LISTEN      107274/java
tcp       0      0  ip:port   ip:port                  ESTABLISHED 107274/java
```

步骤 4 在 AppMaster 启动的节点执行 **telnet 192.168.1.100 23662** 看下是否可以联通该端口，请使用 **root** 用户和 **omm** 用户都执行一遍。 如果出现 **Escape character is '^]'**类似打印则说明可以联通，如果出现 **connection refused** 则表示失败，无法连接到相关端口。

如果相关端口打开，但是从别的节点无法联通到该端口，则需要排查下相关网络配置。

说明

23662 这个端口每次都是随机的，所以要根据自己启动任务打开的端口来测试。

----结束

16.10 提交 Spark 任务时，连接 ResourceManager 异常

问题背景与现象

连接 ResourceManager 异常，导致 Spark 任务提交失败。

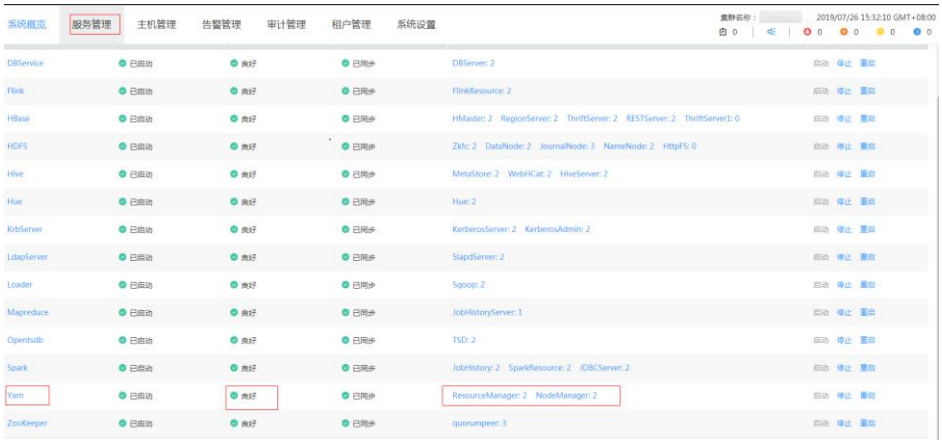
原因分析

1. 在 driver 端打印异常如下，打印连接两个 ResourceManager 主备节点的 26004 端口均被拒绝：

```
15/08/19 18:36:16 INFO RetryInvocationHandler: Exception while invoking
getClusterMetrics of class ApplicationClientProtocolPBClientImpl over 33 after
1 fail over attempts. Trying to fail over after sleeping for 17448ms.
 java.net.ConnectException: Call From ip0 to ip1:26004 failed on connection
exception: java.net.ConnectException: Connection refused.
INFO RetryInvocationHandler: Exception while invoking getClusterMetrics of
class ApplicationClientProtocolPBClientImpl over 32 after 2 fail over attempts.
Trying to fail over after sleeping for 16233ms.
 java.net.ConnectException: Call From ip0 to ip2:26004 failed on connection
exception: java.net.ConnectException: Connection refused;
```

2. 在 MRS Manager 页面查看 ResourceManager 此时是否功能正常，如图 16-3 所示，如果 Yarn 状态故障或某个 yarn 服务的实例出现未知之类的异常说明此时集群的 RM 可能异常。

图16-3 服务状态



3. 排查使用的客户端是否是集群最新的客户端。

   排查集群是否做过实例 RM 迁移相关操作（先卸载某个 RM 实例，然后在其他节点添加回来）。

4. 在 MRS Manager 页面单击“审计管理”，查看审计日志，是否有相关操作的记录。

   使用 **ping** 命令，查看 IP 是否可联通。

解决办法

- 如果 RM 出现异常，可参考 Yarn 相关章节查看解决方法。
- 如果客户端不是最新，请重新下载客户端。
- 若使用 **ping** 命令查看 IP 不通，需要协调网络管理相关人员协助排查网络。
- 若集群开启高可用，尝试将 Yarn 参数“yarn.client.failover-sleep-base-ms”调小。

16.11 DataArts Studio 调度 spark 作业失败

用户问题

DataArts Studio 作业调度失败，显示读取/thriftserver/active_thriftserver 路径下的数据失败。

问题现象

DataArts Studio 作业调度失败，显示读取/thriftserver/active_thriftserver 路径下的数据失败，

报错信息为：Can not get JDBC Connection, due to KeeperErrorCode = NoNode for /thriftserver/active_thriftserver。

原因分析

DataArts Studio 提交 spark 作业时调用 spark 的 JDBC 方式，而 Spark 会启动一个名为 thriftserver 的进程以供客户端提供 JDBC 连接，JDBCServer 在启动时会在 zk 的 /thriftserver 目录下创建子目录 active_thriftserver，并且注册相关连接信息。如果读不到该连接信息就会 JDBC 连接异常。

处理步骤

检查 zookeeper 下面是否有目标目录和注册的信息

步骤 1 以 root 用户登录任意一个 Master 节点并初始化环境变量。

source /opt/client/bigdata_env

步骤 2 执行 **zkCli.sh -server 'ZookeeperIp:2181'**命令登录 zk。

步骤 3 执行 **ls /thriftserver** 查看是否有 active_thriftserver 目录。

- 如果有 active_thriftserver 目录，执行 **get /thriftserver/active_thriftserver** 查看该目录下是否有注册的配置信息。
  - 如果有注册的配置信息，联系支持人员处理。
  - 如果没有注册的配置信息，执行步骤 4
- 如果没有 active_thriftserver 目录，执行步骤 4。

步骤 4 登录 Manager 界面，查看 Spark 的 JDBCServer 实例的主备状态是否未知。

- 是，执行步骤 5。
- 否，联系运维人员处理。

步骤 5 重启两个 JDBCServer 实例，查看主备实例状态恢复正常且 zk 下面有了目标目录和数据，作业即可恢复正常。若实例状态没有恢复请联系支持人员处理。

----结束

16.12 Spark 作业 api 提交状态为 error

用户问题

使用 API 提交 spark 作业后，作业状态显示为 error。

问题类型

作业管理类。

问题现象

修改/opt/client/Spark/spark/conf/log4j.properties 中的日志级别，使用 API V1.1 接口作业提交后，状态显示为 error。

原因分析

executor 会监控作业日志回显，确定作业执行结果，改为 error 后，检测不到输出结果，因此过期后判断作业状态为异常。

处理步骤

将/opt/client/Spark/spark/conf/log4j.properties 中的日志级别修改为 **info**。

建议与总结

建议客户使用 V2 接口提交作业接口。

16.13 集群反复出现 43006 告警

用户问题

集群反复出现“ALM-43006 JobHistory 进程堆内存使用超出阈值”告警，且按照告警参考设置无效。

问题现象

集群出现告警“ALM-43006 JobHistory 进程堆内存使用超出阈值”并且按照指导设置以后，运行一段时间又会出现同样的告警。

原因分析

可能存在 JobHistory 内存泄露问题，需要安装相应的补丁修复。

处理步骤

- 适当调大 JobHistory 进程堆内存。
- 如果已经调大堆内存，可以通过重启 JobHistory 实例规避。

16.14 在 spark-beeline 中创建/删除表失败

用户问题

客户在 spark-beeline 频繁创建和删除大量用户的场景下，个别用户偶现创建/删除表失败。

问题现象

创建表过程：

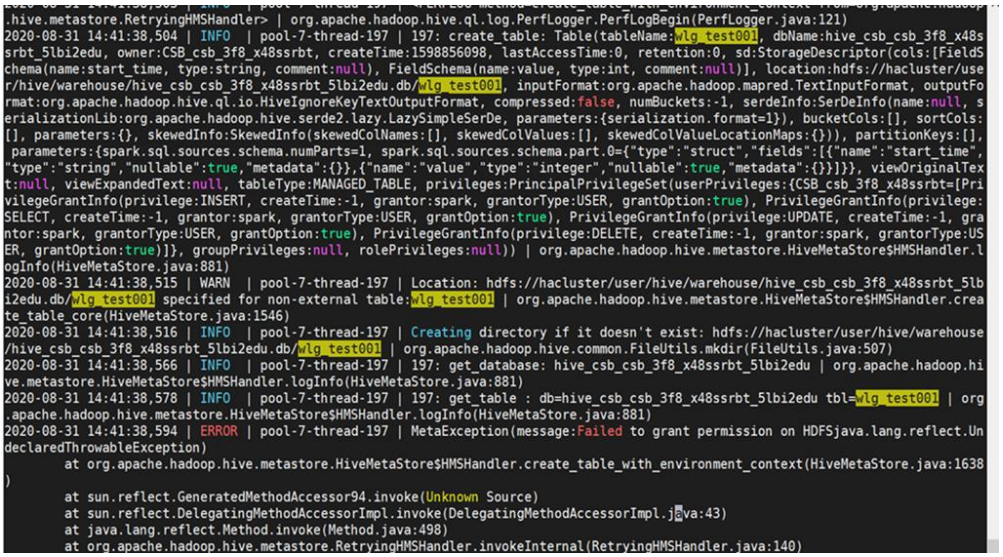
```
CREATE TABLE wlg_test001 (start_time STRING,value INT);
```

报错:

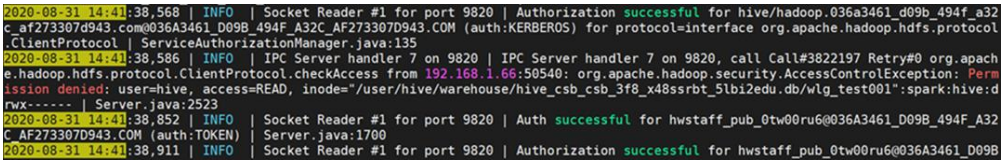
```
Error: org.apache.spark.sql.AnalysisException:
org.apache.hadoop.hive.ql.metadata.HiveException: MetaException(message:Failed to
grant permission on HDFSjava.lang.reflect.UndeclaredThrowableException);
(state=,code=0)
```

原因分析

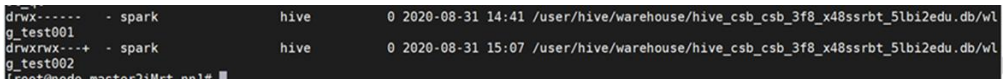
1. 查看 metastore 日志



2. 查看 hdfs 日志



3. 权限对比（test001 为异常用户创建表，test002 为正常用户创建表）

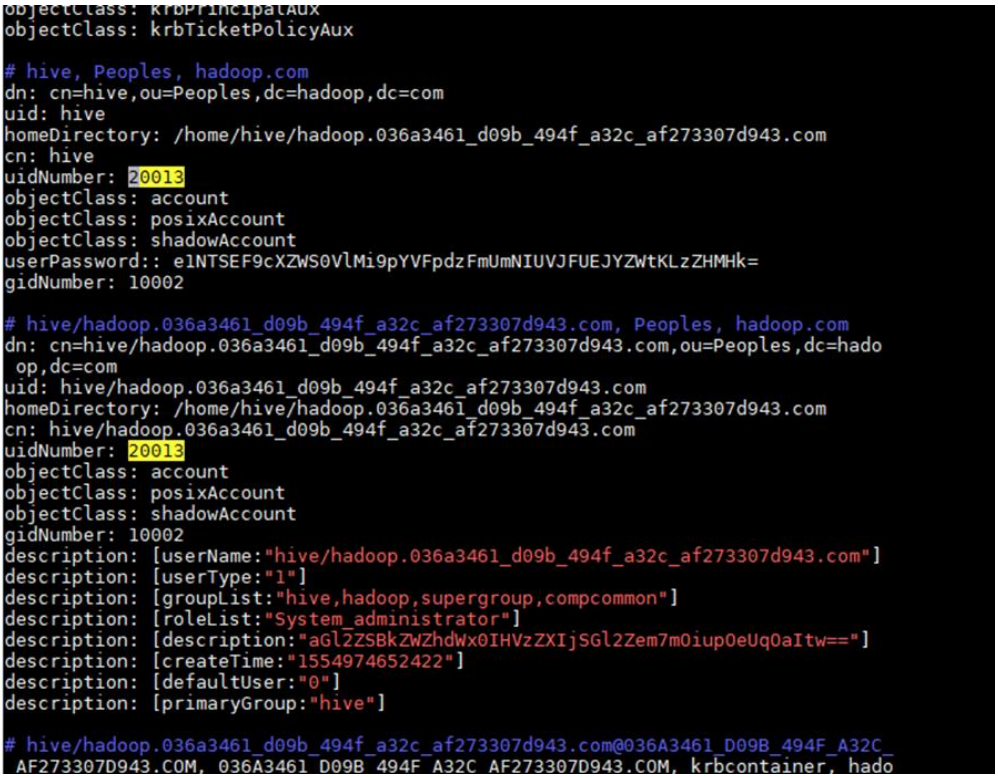


4. drop 表时报类似下面的错

```
0: jdbc:hive2://192.168.1.42:10000/> drop table
dataplan_modela_csbch2;
Error: Error while compiling statement: FAILED:
SemanticException Unable to fetch table dataplan_modela_csbch2.
java.security.AccessControlException: Permission denied:
user=CSB_csb_3f8_x48ssrbt,
access=READ,
inode="/user/hive/warehouse/hive_csb_csb_3f8_x48ssrbt_5lbi2edu.db/dataplan_modela_csbch2":spark:hive:drwx------
```

5. 根因分析。

   创建集群时创建的默认用户使用了相同的 uid，造成用户错乱。在大量创建用户的场景下，触发了该问题，导致在创建表时偶现 hive 用户没有权限。



处理步骤

重启集群 sssd 进程。

以 root 用户执行 **service sssd restart** 命令重启 sssd 服务，执行 **ps -ef | grep sssd** 命令，查看 sssd 进程是否正常。

正常状态为：存在/usr/sbin/sssd 进程和三个子进程/usr/libexec/sssd/sssd_be、/usr/libexec/sssd/sssd_nss、/usr/libexec/sssd/sssd_pam。

16.15 集群外节点提交 Spark 作业到 Yarn 报错连不上 Driver

用户问题

在集群外节点使用 client 模式提交 Spark 任务到 Yarn 上，任务失败，报错为连不上 Driver。

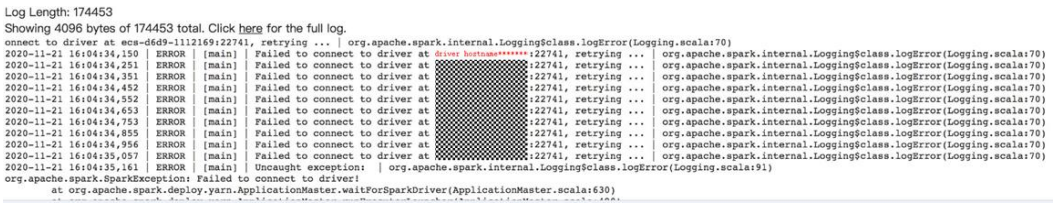
问题现象

集群外节点和集群各个节点网络已经互通，在集群外节点使用 client 模式提交 Spark 任务到 Yarn 上，任务失败，报错为连不上 Driver。

原因分析

使用 client 模式提交 Spark 任务的时候，Spark 的 driver 进程是在客户端这边，而后面的 executor 都需要和 Driver 进行交互来运行作业。

如果 NodeManager 连不上客户端所在的节点，就会报错：



处理步骤

在客户端的 Spark 配置中指定 Driver 的 IP 地址：

“<客户端安装位置>/Spark/spark/conf/spark-defaults.conf”中添加参数“spark.driver.host=driverIP”，重新运行 Spark 任务即可。

建议与总结

建议客户通过 cluster 模式提交作业。

16.16 运行 Spark 任务发现大量 shuffle 结果丢失

用户问题

Spark 任务运行失败，查看任务日志发现大量打印 shuffle 文件丢失。

问题现象

Spark 任务运行失败，查看任务日志发现大量打印 shuffle 文件丢失。

原因分析

spark 运行的时候会将临时产生的 shuffle 文件放在 executor 的临时目录中，方便后面获取。

而当某个 executor 异常退出时，NodeManager 会把这个 executor 所在的 container 临时目录删除，随后其他 executor 再来申请这个 executor 的 shuffle 结果就会报文件找不到。

因此，遇到这样的问题需要确认是否 executor 异常退出，可以根据 spark 任务页面的 executors 便签页查看是否有 dead 状态的 executor，查看各个 dead 状态的 executor 日志，

确认异常退出的原因（其中可能有部分 executor 退出原因就是因为 shuffle 文件找不到，需要找到最早异常退出的 executor）。

常见的异常退出：

- executor 发生 OOM
- executor 运行时出现多个 task 任务失败
- executor 所在节点被清理

处理步骤

根据 executor 异常退出的实际原因调整或者修改任务参数或代码，重新运行 Spark 任务即可。

16.17 JDBCServer 长时间运行导致磁盘空间不足

用户问题

连接 Spark 的 JDBCServer 服务提交 spark-sql 任务到 yarn 集群上，在运行一段时间以后会出现 Core 节点的数据盘被占满的情况。

问题现象

客户连接 Spark 的 JDBCServer 服务提交 spark-sql 任务到 yarn 集群上，在运行一段时间以后会出现 Core 节点的数据盘被占满的情况。

后台查看磁盘使用情况，主要是 JDBCServer 服务的 APP 临时文件（shuffle 生成的文件）太多，并且没有进行清理占用了大量内存。

原因分析

查询 Core 节点有大量文件的目录，发现大部分都是类似“blockmgr-033707b6-fbbb-45b4-8e3a-128c9bcfa4bf”的目录，里面存放了计算过程中产生的 shuffle 临时文件。

因为 JDBCServer 启动了 Spark 的动态资源分配功能，已经将 shuffle 托管给 NodeManager，NodeManager 只会按照 APP 的运行周期来管理这些文件，并不会关注单个 executor 所在的 container 是否存在。因此，只有在 APP 结束的时候才会清理这些临时文件。任务运行时间较长时导致临时文件过多占用了大量磁盘空间。

处理步骤

启动一个定时任务来清理超过一定时间的 shuffle 文件，例如每个整点清理超过 6 个小时的文件：

步骤 1 创建脚本“clean_appcache.sh”，若存在多个数据盘，请根据实际情况修改 BASE_LOC 中 data1 的值。

- 安全集群

```
#!/bin/bash
BASE_LOC=/srv/BigData/hadoop/data1/nm/localdir/usercache/spark/appcache/applica
tion_*/blockmgr*
find $BASE_LOC/ -mmin +360 -exec rmdir {} \;
find $BASE_LOC/ -mmin +360 -exec rm {} \;
```

- 普通集群

```
#!/bin/bash
BASE_LOC=/srv/BigData/hadoop/data1/nm/localdir/usercache/omm/appcache/applicati
on_*/blockmgr*
find $BASE_LOC/ -mmin +360 -exec rmdir {} \;
find $BASE_LOC/ -mmin +360 -exec rm {} \;
```

步骤 2 修改脚本权限。

chmod 755 clean_appcache.sh

步骤 3 增加一个定时任务来启动清理脚本，脚本路径请根据实际脚本存放位置修改。

查看定时任务：crontab -l

编辑定时任务：crontab -e

```
0 * * * * sh /root/clean_appcache.sh > /dev/null 2>&1
```

----结束

16.18 spark-shell 执行 sql 跨文件系统 load 数据到 hive 表失败

用户问题

使用 spark-shell 命令执行 sql 或者 spark-submit 提交的 spark 任务里面有 sql 的 load 命令，并且原数据和目标表存储位置不是同一套文件系统，上述两种方式 MapReduce 任务启动时会报错。

原因分析

当使用 load 导入数据到 hive 表的时候，属于需要跨文件系统的情况（例如原数据在 hdfs 上，而 hive 表数据存放在 obs 上），并且文件长度大于阈值（默认 32M），则会触发使用 distcp 的 MapReduce 任务来执行数据迁移操作。这个 MapReduce 任务配置直接从 spark 任务配置里面提取，但是 spark 任务的 net.topology.node.switch.mapping.impl 配置项不是 hadoop 的默认值，需要使用 spark 的 jar 包，因此 MapReduce 会报类找不到。

处理步骤

方案一：

如果文件较小，则可以将默认长度设置得大于文件最大长度，例如最大的文件是 95M，则设置：

```
hive.exec.copyfile.maxsize=104857600
```

方案二：

如果确实文件较大，需要使用 distcp 任务来提高数据迁移效率，则可以在 spark 任务启动的时候增加设置参数：

```
--conf
spark.hadoop.net.topology.node.switch.mapping.impl=org.apache.hadoop.net.ScriptBaseDMapping
```

16.19 Spark 任务提交失败

问题现象

- Spark 提交任务直接提示无法提交任务。
- Spark 提示无法获取到 yarn 的相关 jar 包。
- 提示多次提交一个文件。

原因分析

- 问题 1：

  最常见的无法提交任务原因是认证失败，

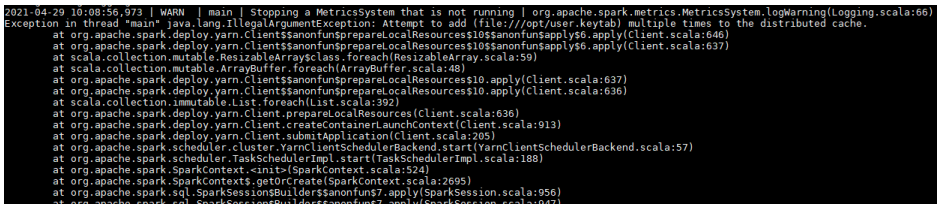
  还有可能是参数设置不正确。

- 问题 2：

  集群默认会把分析节点的 hadoop 相关 jar 包添加到任务的 classpath 中，如果提示 yarn 的包找不到，一般都是因为 hadoop 的相关配置没有设置。

- 问题 3：

  常见的场景是使用--files 上传了 user.keytab，然后使用--keytab 又指定了同一个文件，导致一个文件多次被上传。



处理步骤

- 问题 1：

  重新 kinit 一个用户并修改相应的配置参数。

- 问题 2：

  查看 hadoop 相关的配置项是否正确，查看 spark 的 conf 目录下的 core-site.xml，hdfs-site.xml，yarn-site.xml，mapred-site.xml 等配置文件是否存在问题。

- 问题 3：

重新复制一个 user.keytab，例如：

cp user.keytab user2.keytab

spark-submit --master yarn --files user.keytab --keytab user2.keytab .....

16.20 Spark 任务运行失败

问题现象

- 报错显示 executor 出现 OOM
- 失败的 task 信息显示失败原因是 lost task xxx

原因分析

- 问题 1：一般出现 executor OOM，都是因为数据量过大，也有可能是因为同一个 executor 上面同时运行的 task 太多。
- 问题 2：有些 task 运行失败会报上述错误。当看到这个报错的时候，需要确认的是丢失的这个 task 在哪个节点上面运行，一般的情况是这个丢失的 task 异常退出导致的。

处理步骤

- 问题 1：
  - 对于数据量过大，需要调整 executor 的内存大小的，使用--executor-memory 指定内存大小；
  - 对于同时运行的 task 太多，主要看--executor-cores 设置的 vcore 数量。
- 问题 2：需要在相应的 task 的日志里面查找异常原因。如果有 OOM 的情况，请参照问题 1。

16.21 JDBCServer 连接失败

问题现象

- 提示 ha-cluster 不识别（unknowHost 或者必须加上端口）
- 提示连接 JDBCServer 失败

原因分析

- 问题 1：使用 **spark-beeline** 命令连接 JDBCServer，因为 MRS_3.0 以前的 JDBCServer 是 ha 模式，因此需要使用特定的 url 和 MRS spark 的自带的 jar 包来连接 JDBCServer。
- 问题 2：确认 JDBCServer 服务是否正常，查看对应的端口是否正常监听。

处理步骤

- 问题 1：需要使用特定的 url 和 MRS Spark 的自带的 jar 包来连接 JDBCServer。

- 问题 2：确认 JDBCServer 服务是否正常，查看对应的端口是否正常监听。

16.22 查看 Spark 任务日志失败

问题现象

- 任务运行中查看日志失败
- 任务运行完成，但是查看不到日志

原因分析

- 问题 1：可能原因是 MapReduce 服务异常
- 问题 2：可能原因如下：
  - Spark 的 JobHistory 服务异常。
  - 日志太大，NodeManager 在做日志汇聚的时候出现超时。
  - HDFS 存放日志目录权限异常（默认/tmp/logs/用户名/logs）。
  - 日志已被清理（spark 的 JobHistory 默认存放 7 天的 eventLog，配置项为 spark.history.fs.cleaner.maxAge；MapReduce 默认存放 15 天的任务日志，配置项为 mapreduce.jobhistory.max-age-ms）。
  - 如果 yarn 页面上也找不到，可能是被 yarn 清理了（默认存放 10000 个历史任务，配置项为 yarn.resourcemanager.max-completed-applications）。

处理步骤

- 问题 1：确认 MapReduce 服务是否正常，如果异常，尝试重启服务。如果还是不能恢复，需要查看后台 JobhistoryServer 日志。
- 问题 2：依次排查可能的情况：
  a. 查看 Spark 的 JobHistory 是否运行正常；
  b. 通过查看 yarn 的 app 详情页面，确认日志文件是否过大，如果日志汇聚失败，页面的“Log Aggregation Status:”应该显示为失败或者超时；
  c. 查看对应目录权限是否异常；
  d. 查看目录下是否有对应的 appid 文件（spark 的 eventlog 存放目录：MRS 3.x 及以后版本的目录是 hdfs://hacluster/spark2xJobHistory2x，MRS 3.x 以前版本的目录是 hdfs://hacluster/sparkJobHistory，任务运行日志存放目录是 hdfs://hacluster/tmp/logs/用户名/logs）；
  e. 查看 appid 和当前作业的 id 是否超过历史记录最大值。

16.23 Spark 连接其他服务认证问题

问题现象

- Spark 连接 HBase，报认证失败或者连接不到 hbase 表。

- Spark 连接 HBase 报找不到 jar 包。

原因分析

- 问题 1：HBase 没有获取到当前任务的认证信息，导致连接 HBase 的时候认证失败，无法读取到相应数据
- 问题 2：Spark 默认没有加载 HBase 相关的 jar 包，需要使用--jars 添加到任务中

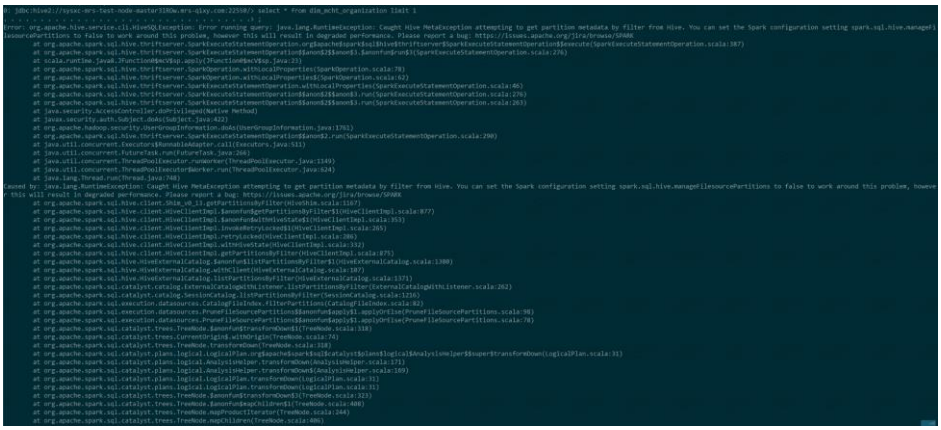
处理步骤

- 问题 1：可以尝试开启 hbase 认证开关：spark.yarn.security.credentials.hbase.enabled=true。但不建议直接用 HBase 客户端的 hbase-site.xml 替换 Spark 客户端下的 hbase-site.xml，两者并不是完全相同。
- 问题 2：需要将 HBase 相关的包使用--jars 上传。

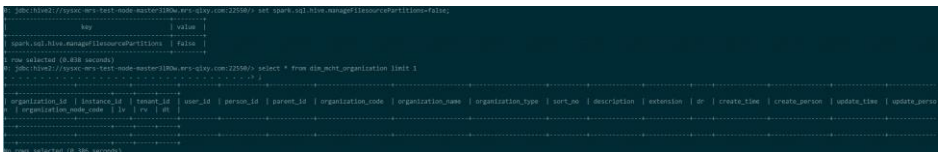
16.24 spark-beeline 查询 Hive 视图报错

用户问题

MRS 3.1.2 版本，spark-beeline 查询 Hive 视图报错，报错如下：



根据提示 set spark.sql.hive.manageFilesourcePartitions=false 后，查不到数据（但是实际在 hive 查询中是有数据的）。



原因分析

转换格式失败，spark-sql 使用其内置的 Metastore，而不是 Hive 中使用的 Metastore，所以读取元数据时失败，spark-sql 内部读取的是 partque 格式的数据，而 Hive 中默认是 orc 的。

处理步骤

步骤 1　登录 Spark 客户端节点，执行如下命令，进入 spark-sql：

cd *{客户端安装目录}*

source bigdata_env

source Spark2x/component_env

kinit *组件业务用户*（普通模式无需执行 kinit）

spark-sql

步骤 2　执行如下命令设置 spark.sql.hive.convertMetastoreOrc=false。

set spark.sql.hive.convertMetastoreOrc=false;

步骤 3　重新查询 Hive 视图，显示正常。

----结束

17 使用 Sqoop

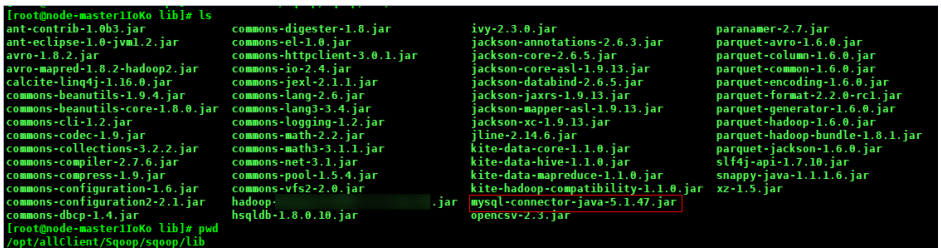
17.1 Sqoop 如何连接 mysql

用户问题

Sqoop 如何连接 mysql。

处理步骤

步骤 1 在集群上安装客户端，查看客户端 sqoop/lib 下是否有 mysql 驱动包。



步骤 2 在客户端目录下加载环境变量。

source bigdata_env

步骤 3 Kerberos 认证。

如果集群已启用 Kerberos 认证，执行以下命令认证当前用户。如果当前集群未启用

跳过此步骤。

kinit MRS 集群用户

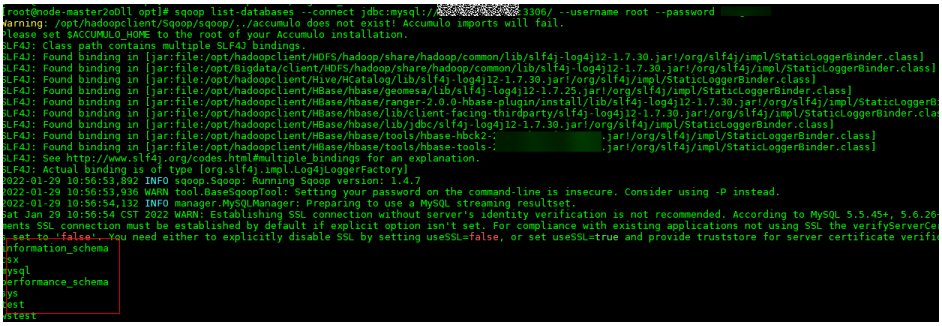
例如：

kinit admin

步骤 4 连接数据库。

sqoop list-databases --connect jdbc:mysql://IP:3306/ --username 用户名 --password 密码

如下：



上图所示：则代表 sqoop 连接 mysql 成功。

----结束

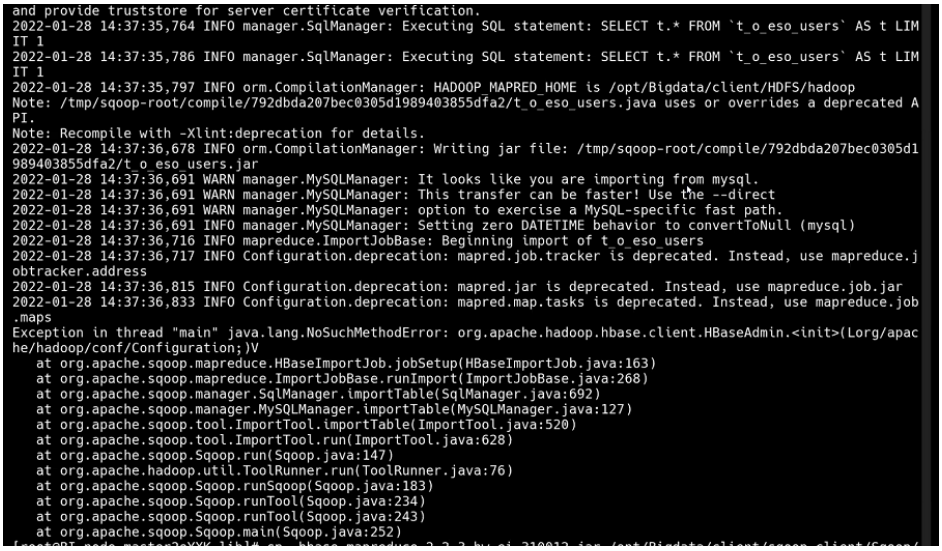
17.2 Sqoop 读取 MySQL 中数据到 HBase 报 HBaseAdmin.<init>方法找不到异常

问题

使用 MRS 的 Sqoop 客户端（1.4.7 版本），从 MySQL 数据库中指定表抽取数据，存放到 HBase（2.2.3 版本）指定的表中，报出异常：

```
Trying to load data into HBASE through Sqoop getting below error.
Exception in thread "main" java.lang.NoSuchMethodError:
org.apache.hadoop.hbase.client.HBaseAdmin.<init>(Lorg/apache/hadoop/conf/Configurat
ion;)V
```

完整异常信息如图所示：



执行 Sqoop 抽取数据命令样例：

```
sqoop import \
--connect jdbc:mysql://mysql 服务器地址:端口号/database1 \
--username admin \
```

```
--password xxx \
--table table1 \
--hbase-table table2 \
--column-family info \
--hbase-row-key id \
--hbase-create-table --m 1
```

处理步骤

Sqoop 客户端安装完成之后，没有直接引入 HBase 相关的依赖 jar 包，需要通过手动导入指定低版本的 HBase 相关依赖 jar 包。解决方法步骤如下：

步骤 1 确认 Sqoop 客户端和 HBase 客户端是否在同一个路径下。

- 是，执行步骤 2。
- 否，删除原有的 Sqoop 和 HBase 客户端文件，从 FusionInsight Manager 上下载完整的客户端安装在同一路径下。执行步骤 2。

步骤 2 以 **root** 用户登录 Sqoop 客户端安装节点。

步骤 3 下载 HBase 1.6.0 版本的 jar 包上传到 Sqoop 客户端的“lib”目录下。

步骤 4 上传包之后，修改包的权限，可以设置为 755，具体执行命令为：

chmod 755 *包名称*

步骤 5 在客户端目录下执行以下命令刷新 Sqoop 客户端：

source bigdata_env

重新执行 sqoop 命令

----结束

17.3 HUE 界面的 Sqoop 任务 HBase 到 HDFS 报错

本章节仅适用于 MRS 1.9.2 版本集群。

用户问题

利用 HUE 的 sqoop 操作把 HBase 中的数据导入 HDFS 中报错。

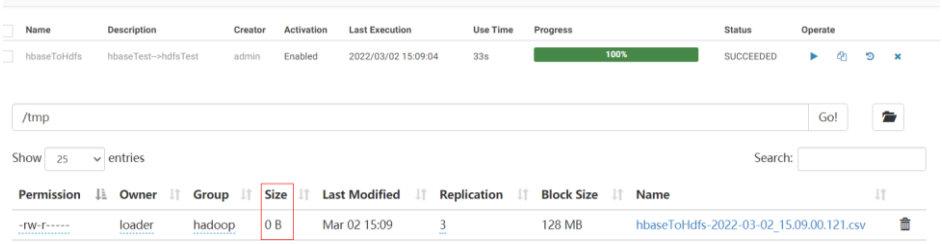
Caused by: java.lang.ClassNotFoundException: org.apache.htrace.Trace

```
2022-03-02 15:09:00,264 [main] ERROR org.apache.sqoop.connector.hbase.HBaseExtractor  - An exceptional condition has occurred.
org.apache.sqoop.common.SqoopException: HBASE_CONNECTOR_0011:Failed to open table.
        at org.apache.sqoop.connector.hbase.HBaseExtractor.openDB(HBaseExtractor.java:239)
        at org.apache.sqoop.connector.hbase.HBaseExtractor.access$100(HBaseExtractor.java:34)
        at org.apache.sqoop.connector.hbase.HBaseExtractor$1.run(HBaseExtractor.java:86)
        at org.apache.sqoop.connector.hbase.HBaseExtractor$1.run(HBaseExtractor.java:76)
        at org.apache.sqoop.connector.hbase.HBaseExtractor.extract(HBaseExtractor.java:114)
        at org.apache.sqoop.connector.hbase.HBaseExtractor.extract(HBaseExtractor.java:34)
        at org.apache.sqoop.job.mr.SqoopMapper.runInternal(SqoopMapper.java:156)
        at org.apache.sqoop.job.mr.SqoopMapper.run(SqoopMapper.java:79)
        at org.apache.hadoop.mapred.MapTask.runNewMapper(MapTask.java:787)
        at org.apache.hadoop.mapred.MapTask.run(MapTask.java:341)
        at org.apache.hadoop.mapred.YarnChild$2.run(YarnChild.java:188)
        at java.security.AccessController.doPrivileged(Native Method)
        at javax.security.auth.Subject.doAs(Subject.java:422)
        at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1840)
        at org.apache.hadoop.mapred.YarnChild.main(YarnChild.java:182)
Caused by: java.lang.reflect.InvocationTargetException
```

```
Caused by: java.lang.reflect.InvocationTargetException
        at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
        at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:62)
        at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:45)
        at java.lang.reflect.Constructor.newInstance(Constructor.java:423)
        at org.apache.hadoop.hbase.client.ConnectionFactory.createConnection(ConnectionFactory.java:238)
        at org.apache.hadoop.hbase.client.ConnectionManager.createConnection(ConnectionManager.java:454)
        at org.apache.hadoop.hbase.client.ConnectionManager.createConnection(ConnectionManager.java:447)
        at org.apache.hadoop.hbase.client.ConnectionManager.getConnectionInternal(ConnectionManager.java:325)
        at org.apache.hadoop.hbase.client.HTable.<init>(HTable.java:184)
        at org.apache.hadoop.hbase.client.HTable.<init>(HTable.java:150)
        at org.apache.sqoop.connector.hbase.HBaseExtractor.openDB(HBaseExtractor.java:236)
        ... 14 more
Caused by: java.lang.NoClassDefFoundError: org/apache/htrace/Trace
        at org.apache.hadoop.hbase.zookeeper.RecoverableZooKeeper.exists(RecoverableZooKeeper.java:245)
        at org.apache.hadoop.hbase.zookeeper.ZKUtil.checkExists(ZKUtil.java:436)
        at org.apache.hadoop.hbase.zookeeper.ZKClusterId.readClusterIdZNode(ZKClusterId.java:65)
        at org.apache.hadoop.hbase.client.ZooKeeperRegistry.getClusterId(ZooKeeperRegistry.java:105)
        at org.apache.hadoop.hbase.client.ConnectionManager$HConnectionImplementation.retrieveClusterId(ConnectionManager.java:944)
        at org.apache.hadoop.hbase.client.ConnectionManager$HConnectionImplementation.<init>(ConnectionManager.java:720)
        ... 25 more
Caused by: java.lang.ClassNotFoundException: org.apache.htrace.Trace
        at java.net.URLClassLoader.findClass(URLClassLoader.java:382)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:419)
```

问题现象

Sqoop 任务运行成功，但 hdfs 中的 csv 文件无内容。



原因分析

推测 jar 包冲突或者缺少 jar 包造成的。

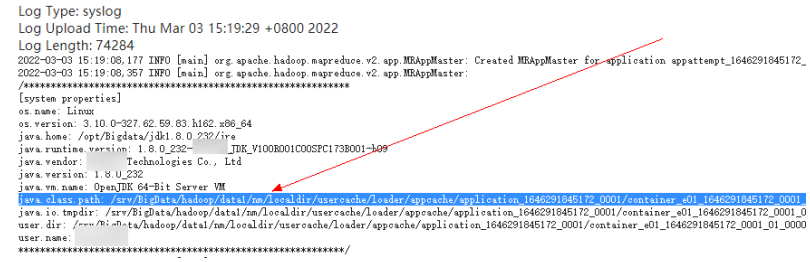
处理步骤

步骤 1　去 sqoop 的 lib 下 grep。

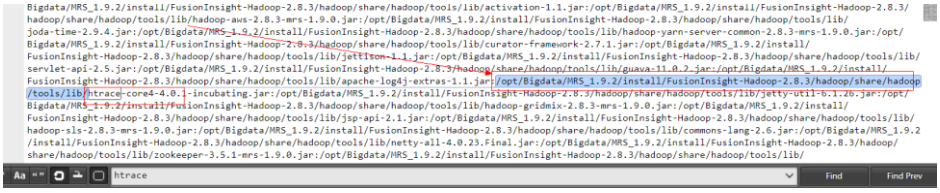
1. 进入 sqoop 的 lib 目录下，进行 grep 查找。

```
[root@node-master1PMPi lib]# pwd
/opt/Bigdata/MRS_1.9.2/install/FusionInsight-Sqoop-1.99.7/FusionInsight-Sqoop-1.99.7/server/lib
[root@node-master1PMPi lib]# grep org.apache.htrace.Trace *
Binary file htrace-core-3.1.0-incubating.jar matches
[root@node-master1PMPi lib]#
```

2. 进入 yarn 原生界面，查看运行的任务的报错具体信息。



3. 将 java.class.path 复制出来，搜索 htrace-core。



4. 复制 jar 包到如下位置。

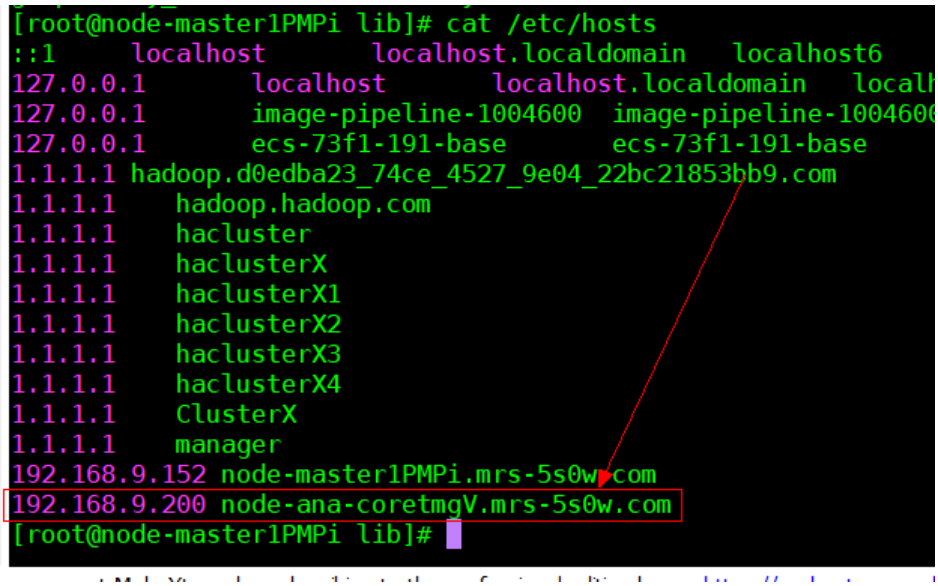
   cp /opt/Bigdata/MRS_1.9.2/install/FusionInsight-Sqoop-1.99.7/FusionInsight-Sqoop-1.99.7/server/lib/htrace-core-3.1.0-incubating.jar /opt/Bigdata/MRS_1.9.2/install/FusionInsight-Hadoop-2.8.3/hadoop/share/hadoop/common/lib/

5. 修改权限。

   chmod 777 *htrace-core-3.1.0-incubating.jar* （真实复制的 jar 包）

   chown omm:ficommon *htrace-core-3.1.0-incubating.jar* （真实复制的 jar 包）

6. 查看 hosts 文件，对其他所有节点进行同样的复制 jar 包操作。



7. 重新运行 sqoop 任务，产生报错如下：

```
        at java.lang.Thread.run(Thread.java:745)
Caused by: com.google.protobuf.ServiceException: java.lang.NoClassDefFoundError: com/yammer/metrics/core/Gauge
        at org.apache.hadoop.hbase.ipc.AbstractRpcClient.callBlockingMethod(AbstractRpcClient.java:240)
        at org.apache.hadoop.hbase.ipc.AbstractRpcClient$BlockingRpcChannelImplementation.callBlockingMethod(AbstractRp
        at org.apache.hadoop.hbase.protobuf.generated.ClientProtos$ClientService$BlockingStub.scan(ClientProtos.java:3E
        at org.apache.hadoop.hbase.client.ClientSmallReversedScanner$SmallReversedScannerCallable.call(ClientSmallRever
        ... 9 more
Caused by: java.lang.NoClassDefFoundError: com/yammer/metrics/core/Gauge
        at org.apache.hadoop.hbase.ipc.AbstractRpcClient.callBlockingMethod(AbstractRpcClient.java:225)
        ... 12 more
Caused by: java.lang.ClassNotFoundException: com.yammer.metrics.core.Gauge
        at java.net.URLClassLoader.findClass(URLClassLoader.java:382)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:419)
        at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:352)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:352)
        ... 13 more
2022-03-03 15:45:01,714 [main] INFO  org.apache.sqoop.job.mr.SqoopMapper  - Extractor has finished
2022-03-03 15:45:01,715 [main] INFO  org.apache.sqoop.job.mr.SqoopMapper  - Stopping progress service
2022-03-03 15:45:01,727 [main] INFO  org.apache.sqoop.job.mr.SqoopOutputFormatLoadExecutor  - SqoopOutputFormatLoadExec
2022-03-03 15:45:01,776 [OutputFormatLoader-consumer] INFO  org.apache.sqoop.job.mr.SqoopOutputFormatLoadExecutor  - Lo
2022-03-03 15:45:01,777 [main] INFO  org.apache.sqoop.job.mr.SqoopOutputFormatLoadExecutor  - SqoopOutputFormatLoadExec

Log Type: stdout
Log Upload Time: Thu Mar 03 15:45:15 +0800 2022
Log Length: 0

Log Type: syslog
```

步骤 2　去 hbase 的 lib 下 grep。

1. 进入 hbase 的 lib 目录下，进行 grep 查找。

```
[root@node-master1PMPi lib]# pwd
/opt/Bigdata/MRS_1.9.2/install/FusionInsight-HBase-1.3.1/hbase/lib
[root@node-master1PMPi lib]# grep com.yammer.metrics.core.Gauge *
grep: jline: Is a directory
Binary file metrics-core-2.2.0.jar matches
grep: native: Is a directory
grep: ruby: Is a directory
grep: ruby_luna: Is a directory
[root@node-master1PMPi lib]#
```

2. 继续复制 jar 包过去。

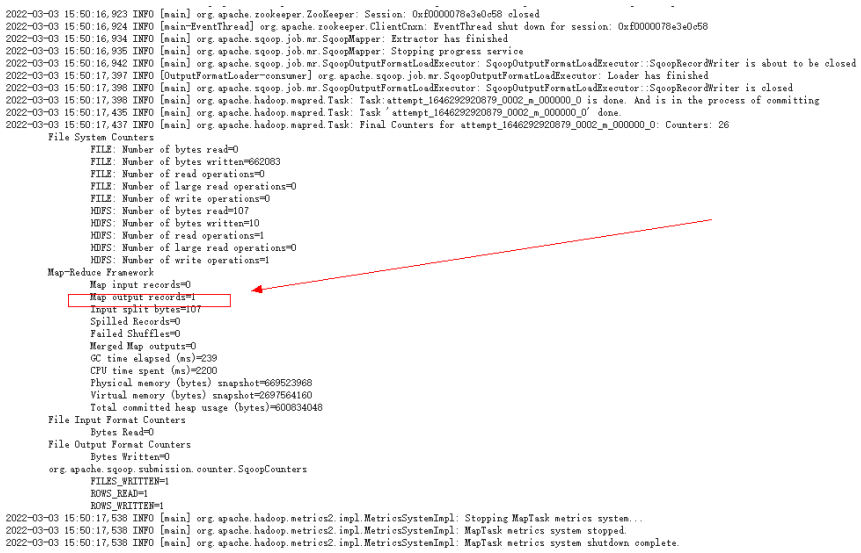
   cp /opt/Bigdata/MRS_1.9.2/install/FusionInsight-HBase-1.3.1/hbase/lib/metrics-core-2.2.0.jar /opt/Bigdata/MRS_1.9.2/install/FusionInsight-Hadoop-2.8.3/hadoop/share/hadoop/common/lib/

3. 修改文件权限。

   chmod 777 *metrics-core-2.2.0.jar*　（真实复制的 jar 包）

   chown omm:ficommon *metrics-core-2.2.0.jar*（真实复制的 jar 包）

4. 查看 hosts 文件，对其他所有节点进行同样的复制 jar 包操作。
5. 继续运行 sqoop 任务，成功。



----结束

处理总结

1. 将 sqoop 的 lib 下 htrace-core-3.1.0-incubating.jar 和 hbase 的 lib 下的 metrics-core-2.2.0.jar，复制到/opt/Bigdata/MRS_1.9.2/install/FusionInsight-Hadoop-2.8.3/hadoop/share/hadoop/common/lib/下。
2. 修改 jar 包的文件权限为 777 和 omm:ficommon。
3. 所有节点均采取以上操作，重新运行 sqoop 任务即可。

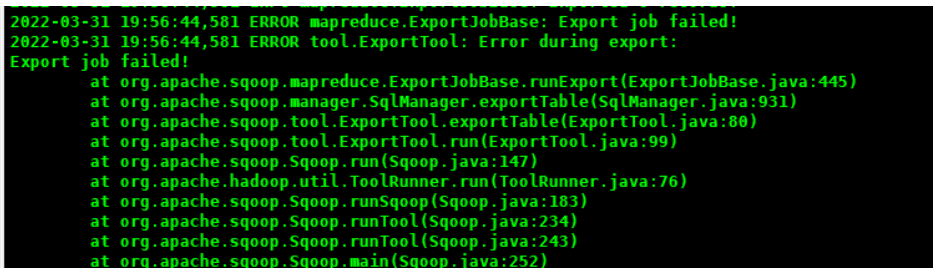
17.4 Sqoop 从 hive 到 mysql8.0 报格式错误

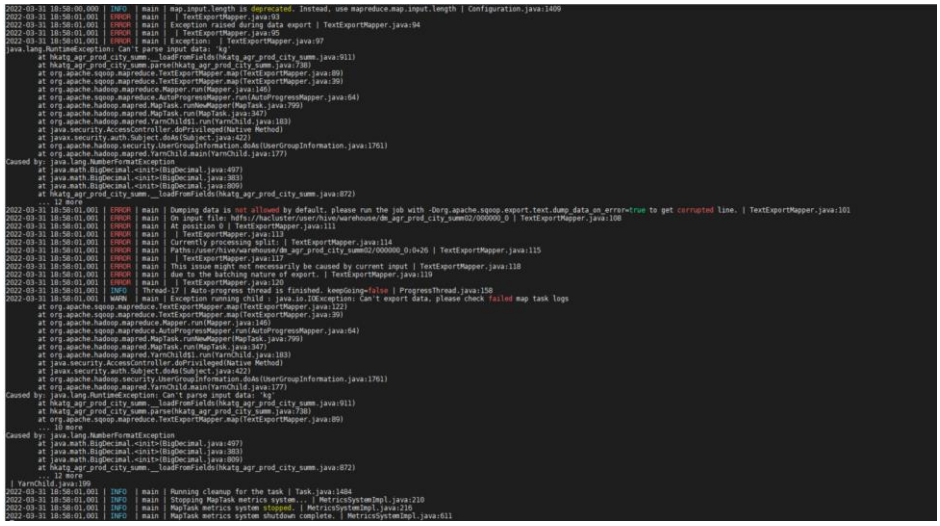
本章节仅适用于 MRS 3.1.0 版本集群。

用户问题

310 集群 Sqoop 从 hive 到 mysql8.0 报格式错误。

问题现象





原因分析

通过日志判断得知是格式异常。

处理步骤

确认分隔符、表字段的格式无问题，在 sqoop 语句中添加--columns 绑定对应字段。

sqoop export --connect jdbc:mysql://IP:端口号/数据库名 --username 用户名 --password 密码 --table 表名 --columns 列字段（多个列用英文逗号分开） -export-dir 导出地址 --fields-terminated-by 分隔符 --input-null-string '\\N' --input-null-non-string '\\N' -m 1

样例：

sqoop export --connect jdbc:mysql://172.16.0.6:3306/lidengpeng --username root --password Mrs@2021 --table hkatg_agr_prod_city_summ --columns year,city_name,city_code,prod_code,prod_name,prod_type,sown_area,area_unit,yield_wegt,yield_unit,total_wegt,total_wegt_unit,data_sorc_code,etl_time -export-dir hdfs://hacluster/user/hive/warehouse/dm_agr_prod_city_summ02 --fields-terminated-by ',' --input-null-string '\\N' --input-null-non-string '\\N' -m 1

17.5 Sqoop import 从 pg 到 hive 报错

背景

使用 sqoop import 命令抽取开源 postgre 到 MRS hdfs 或 hive 等。

用户问题

使用 sqoop 命令查询 postgre 表可以，但是执行 sqoop import 命令导入导出时报错：

The authentication type 5 is not supported. Check that you have configured the pg_hba.conf file to include the client's IP address or subnet。

原因分析

1. 连接 postgresql MD5 认证不通过，需要在 pg_hba.cnf 配置白名单。
2. 在执行 sqoop import 命令时，会启动 MapReduce 任务，由于 MRS Hadoop 安装目 /opt/Bigdata/FusionInsight_HD_*/1_*_DataNode/install/hadoop/share/hadoop/common/lib 下自带了 postgre 驱动包 gsjdbc4-*.jar，与开源 postgre 服务不兼容导致报错。

处理步骤

1. 客户在 pg_hba.cnf 配置白名单。
2. 驱动重复，集群自带，将其余驱动排除出去，所有 core 节点上的 gsjdbc4jar 包去掉，在 sqoop/lib 下添加 postgrejar 包即可。

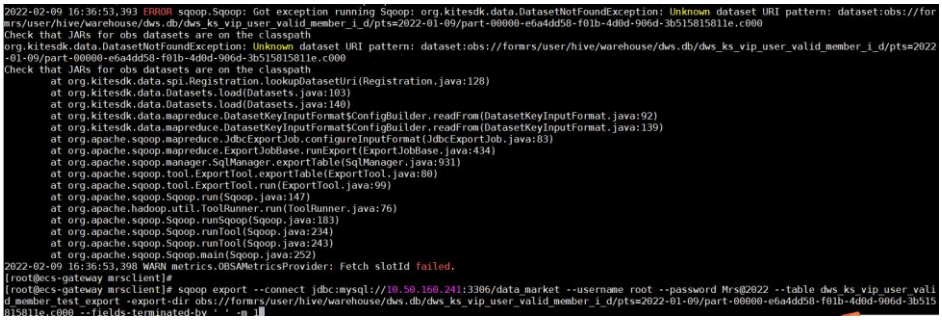
   mv /opt/Bigdata/FusionInsight_HD_*/1_*_DataNode/install/hadoop/share/hadoop/common/lib/gsjdbc4-*.jar /tmp

17.6 Sqoop 读 mysql，写 parquet 文件到 OBS 失败

用户问题

sqoop 读 mysql 数据，然后直接写到 obs，指定 parquet 格式时写入报错，不指定 parquet 时不报错。

问题现象



原因分析

parquet 不支持 hive3，用 Hcatalog 方式写入。

处理步骤

采用 Hcatalog 的方式：参数指定对应的 hive 库和表，需要修改 SQL 语句指定到具体字段（需要客户修改脚本）。

具体如下：

客户原来的脚本：

sqoop import --connect 'jdbc:mysql://10.160.5.65/xxx_pos_online_00?zeroDateTimeBehavior=convertToNull' --username root --password Mrs@2022

--split-by id

--num-mappers 2

--query 'select * from pos_remark where 1=1 and $CONDITIONS'

--target-dir obs://za-test/dev/xxx_pos_online_00/pos_remark

--delete-target-dir

--null-string '\\N‘

--null-non-string '\\N‘

--as-parquetfile

修改后的脚本（可以执行成功）：

sqoop import --connect 'jdbc:mysql://10.160.5.65/xxx_pos_online_00?zeroDateTimeBehavior=convertToNull' --username root --password Mrs@2022

--split-by id

--num-mappers 2

--query 'select id,pos_case_id,pos_transaction_id,remark,update_time,update_user,is_deleted,creator,modifier,gmt_created,gmt_modified,update_user_id,tenant_code from pos_remark where 1=1 and $CONDITIONS‘

--hcatalog-database xxx_dev

--hcatalog-table ods_pos_remark

17.7 Sqoop 迁移数据库数据报错

用户问题

- MRS Sqoop 迁移数据库数据报错。

  报错如：Communications link failure;

  The driver has not received any packets from the server；
- 源端为 dws 时报错如：got exception running sqoop .java.lang.Runtime.Exception, could not load db driver class。

图17-1 Sqoop 迁移数据库数据报错

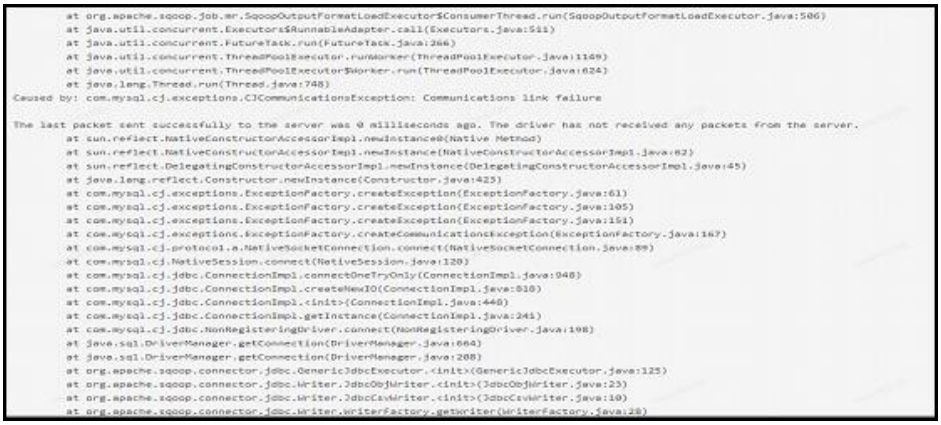
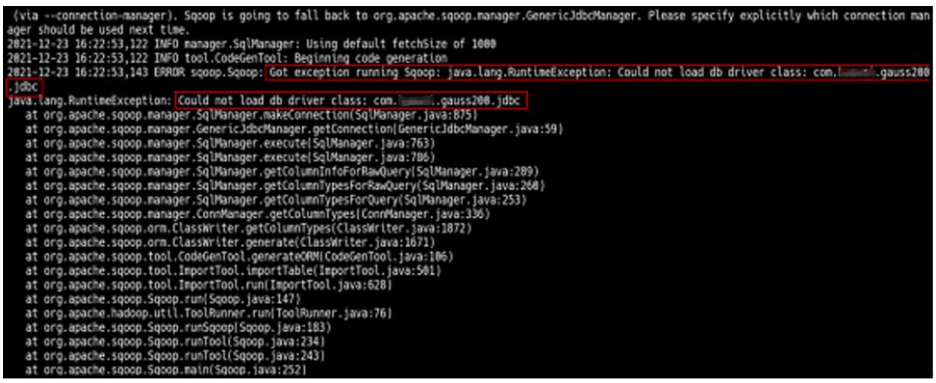


图17-2 源端为 dws 时报错



问题分析

从上述两个报错分析，有可能是驱动问题，服务端未正确加载 jdbc 驱动，或者是网络原因，由于 sqoop 迁移数据过程，map reduce 任务是运行在分析 core 节点，因此需要保证正常加载驱动类的情况下，保证源端数据库和集群节点的网络正常通信。

处理步骤

1. 检查源端数据库端口对集群节点均放通。

   若使用 EIP 公网通信，那么集群节点均需要绑定公网，尤其是分析节点，必须绑定公网才能保证作业正常运行。

   说明

   sqoop 实例一般分布在 master 节点，但作业真正运行在分析节点，因此创建数据库连接，是和 master 节点通信，

   作业是通过 Yarn 任务分发到分析节点执行。

2. 检查 sqoop 实例安装目录下是否放置数据库驱动 jar 包，并保证 jar 包路径正确，文件权限及属主属组正确；配置文件正确配置；保证这些后需要重启 sqoop 实例后才可以正常加载驱动。
   - MRS 3.X 集群驱动路径为：/opt/Bigdata/FusionInsight_Current/1_xx_LoaderServer/install/FusionInsight-Sqoop-1.99.3/server/webapps/loader/WEB-INF/ext-lib/
   - MRS3.X 之前版本集群驱动路径为：/opt/Bigdata/MRS_XXX/install/FusionInsight-Sqoop-1.99.7/FusionInsight-Sqoop-1.99.7/server/jdbc/

   说明

   3.x 之前版本需要修改配置文件。

3. 如果是命令行提交作业，建议指定 ：--driver 参数。

   如源端是 dws 时命令行加：--driver com.xxx.gauss200.jdbc.Driver。

   其他数据库根据实际情况填写--driver 参数。

18 使用 Storm

18.1 Storm 组件的 Storm UI 页面中 events 超链接地址无效

用户问题

Storm 组件的 Storm UI 页面中 events 超链接地址无效。

问题现象

用户提交拓扑后无法查看拓扑数据处理日志，按钮 events 地址无效。

原因分析

MRS 集群提交拓扑时默认不开启拓扑数据处理日志查看功能。

处理步骤

步骤 1 登录 Storm WebUI：

- MRS 2.x 及之前版本：选择“Storm”，在“Storm 概述”的“Storm WebUI”，单击任意一个 UI 链接，打开 Storm 的 WebUI。

  说明

  第一次访问 Storm WebUI，需要在浏览器中添加站点信任以继续打开页面。

- MRS 3.x 及后续版本：选择“Storm > 概览”，在“基本信息”的“Storm WebUI”，单击任意一个 UI 链接，打开 Storm 的 WebUI。

步骤 2 单击“Topology Summary”区域的指定拓扑名称，打开拓扑的详细信息。

步骤 3 在“Topology actions”区域单击“Kill”删除已经提交的 Storm 拓扑。

步骤 4 重新提交 Storm 拓扑，并开启查看拓扑数据处理日志功能，在提交 Storm 拓扑时增加参数“topology.eventlogger.executors”，该参数设置为一个不为 0 的正整数。例如：

storm jar ***拓扑包路径 拓扑 Main 方法的类名称 拓扑名称*** **-c topology.eventlogger.executors=X**

步骤 5 在 Storm UI 界面，单击“Topology Summary”区域的指定拓扑名称，打开拓扑的详细信息。

步骤 6 在“Topology actions”区域单击“Debug”，输入采样数据的百分比数值，并单击“OK”开始采样。

步骤 7 单击拓扑的“Spouts”或“Bolts”任务名称，在“Component summary”单击“events”即可打开处理数据日志。

说明

如需开启特定“Spouts”或“Bolts”任务的拓扑数据处理日志查看功能，请单击拓扑的“Spouts”或“Bolts”任务名称后，“Topology actions”区域单击“Debug”按钮，输入采样数据的百分比数值。

----结束

18.2 提交拓扑失败

问题背景与现象

使用 MRS 流式集群，主要安装 ZooKeeper、Storm、Kafka。

使用客户端命令，提交 Topology 失败。

可能原因

- Storm 服务异常。
- 客户端用户没有进行安全认证或者认证过期。
- 提交拓扑中包含 storm.yaml 文件和服务端冲突。

原因分析

用户提交拓扑失败，可能原因客户端侧问题或者 Storm 侧问题。

1. 查看 Storm 状态。

   MRS Manager：

   登录 MRS Manager，在 MRS Manager 页面，选择“服务管理 > Storm”，查看 Storm 服务当前状态，发现状态为“良好”，且监控指标内容显示正确。

   FusionInsight Manager 界面操作：

   对于 MRS 3.x 及后续版本集群：登录 FusionInsight Manager。选择“集群 > 服务 > Storm”，查看 Storm 服务当前状态，发现状态为“良好”，且监控指标内容显示正确。

2. 查看客户端提交日志，发现打印 KeeperExceptionSessionExpireException 异常信息，如下所示：

```
org.apache.zookeeper.KeeperException$SessionExpiredException: KeeperErrorCode = Session expired
        at org.apache.zookeeper.KeeperException.create(KeeperException.java:131) ~[zookeeper-3.5.0.jar:3.5.0-V100R002C00B109]
        at org.apache.curator.framework.imps.CuratorFrameworkImpl.checkBackgroundRetry(CuratorFrameworkImpl.java:710) [curator-framework-2.5.0.jar:na]
        at org.apache.curator.framework.imps.CuratorFrameworkImpl.processBackgroundOperation(CuratorFrameworkImpl.java:510) [curator-framework-2.5.0.jar:na]
        at org.apache.curator.framework.imps.BackgroundSyncImpl$1.processResult(BackgroundSyncImpl.java:50) [curator-framework-2.5.0.jar:na]
        at org.apache.zookeeper.ClientCnxn$EventThread.processEvent(ClientCnxn.java:684) [zookeeper-3.5.0.jar:3.5.0-V100R002C00B109]
        at org.apache.zookeeper.ClientCnxn$EventThread.queuePacket(ClientCnxn.java:498) [zookeeper-3.5.0.jar:3.5.0-V100R002C00B109]
        at org.apache.zookeeper.ClientCnxn.finishPacket(ClientCnxn.java:731) [zookeeper-3.5.0.jar:3.5.0-V100R002C00B109]
        at org.apache.zookeeper.ClientCnxn.conLossPacket(ClientCnxn.java:765) [zookeeper-3.5.0.jar:3.5.0-V100R002C00B109]
        at org.apache.zookeeper.ClientCnxn.access$2700(ClientCnxn.java:97) [zookeeper-3.5.0.jar:3.5.0-V100R002C00B109]
        at org.apache.zookeeper.ClientCnxn$SendThread.cleanup(ClientCnxn.java:1391) [zookeeper-3.5.0.jar:3.5.0-V100R002C00B109]
        at org.apache.zookeeper.ClientCnxn$SendThread.run(ClientCnxn.java:1314) [zookeeper-3.5.0.jar:3.5.0-V100R002C00B109]
2016-05-31 09:25:24 | INFO  | [main] | Session: 0x10027394760ab4b closed | org.apache.zookeeper.ZooKeeper (ZooKeeper.java:968)
Exception in thread "main" java.lang.RuntimeException: Exception while initializing NimbusLeaderElections
        at backtype.storm.nimbus.NimbusLeaderElections.init(NimbusLeaderElections.java:86)
        at backtype.storm.utils.NimbusClient.getConfiguredClient(NimbusClient.java:39)
        at backtype.storm.StormSubmitter.submitTopology(StormSubmitter.java:169)
        at backtype.storm.StormSubmitter.submitTopologyWithProgressBar(StormSubmitter.java:256)
        at backtype.storm.StormSubmitter.submitTopologyWithProgressBar(StormSubmitter.java:236)
        at storm.starter.WordCountTopology.main(WordCountTopology.java:94)
Caused by: org.apache.zookeeper.KeeperException$ConnectionLossException: KeeperErrorCode = ConnectionLoss for /storm/nimbus-leader
        at org.apache.zookeeper.KeeperException.create(KeeperException.java:99)
        at org.apache.zookeeper.KeeperException.create(KeeperException.java:51)
        at org.apache.zookeeper.ZooKeeper.exists(ZooKeeper.java:1501)
        at org.apache.curator.framework.imps.ExistsBuilderImpl$2.call(ExistsBuilderImpl.java:172)
        at org.apache.curator.framework.imps.ExistsBuilderImpl$2.call(ExistsBuilderImpl.java:161)
        at org.apache.curator.RetryLoop.callWithRetry(RetryLoop.java:107)
        at org.apache.curator.framework.imps.ExistsBuilderImpl.pathInForeground(ExistsBuilderImpl.java:157)
        at org.apache.curator.framework.imps.ExistsBuilderImpl.forPath(ExistsBuilderImpl.java:148)
        at org.apache.curator.framework.imps.ExistsBuilderImpl.forPath(ExistsBuilderImpl.java:36)
        at backtype.storm.nimbus.NimbusLeaderElections.init(NimbusLeaderElections.java:66)
        ... 5 more
```

上述错误是由于在提交拓扑之前没有进行安全认证或者认证后 TGT 过期导致。

解决方法参考步骤 1。

3. 查看客户端提交日志，发现打印 ExceptionInIntializerError 异常信息，提示 Found multiple storm.yaml resources。如下所示：

```
Exception in thread "main" java.lang.ExceptionInInitializerError
        at backtype.storm.topology.TopologyBuilder.createTopology(TopologyBuilder.java:106)
        at com.huawei.streaming.storm.example.wordcount.WordCountTopology.cmdSubmit(WordCountTopology.java:117)
        at com.huawei.streaming.storm.example.wordcount.WordCountTopology.submitTopology(WordCountTopology.java:80)
        at com.huawei.streaming.storm.example.wordcount.WordCountTopology.main(WordCountTopology.java:71)
Caused by: java.lang.RuntimeException: Found multiple storm.yaml resources. You're probably bundling the Storm jars with your topology jar.
        at backtype.storm.utils.Utils.findAndReadConfigFile(Utils.java:151)
        at backtype.storm.utils.Utils.readStormConfig(Utils.java:206)
        at backtype.storm.utils.Utils.<clinit>(Utils.java:70)
        ... 4 more
```

该错误是由于业务 jar 包中存在 storm.yaml 文件，和服务端的 storm.yaml 文件冲突导致的。

解决方法参考步骤 2。

4. 如果不是上述原因，则请参考 18.3 提交拓扑失败，提示 Failed to check principle for keytab。

解决办法

步骤 1 认证异常。

1. 登录客户端节点，进入客户端目录。
2. 执行以下命令重新提交任务。（业务 jar 包和 Topology 根据实际情况替换）

   source bigdata_env

   kinit *用户名*

   storm jar storm-starter-topologies-0.10.0.jar storm.starter.WordCountTopology test

步骤 2 拓扑包异常。

排查业务 jar，将业务 jar 中 storm.yaml 文件删除，重新提交任务。

----结束

18.3 提交拓扑失败，提示 Failed to check principle for keytab

问题背景与现象

使用 MRS 流式安全集群，主要安装 ZooKeeper、Storm、Kafka 等。

定义拓扑访问 HDFS、HBase 等组件，使用客户端命令，提交 Topology 失败。

可能原因

- 提交拓扑中没有包含用户的 keytab 文件。
- 提交拓扑中包含的 keytab 和提交用户不一致。
- 客户端/tmp 目录下已存在 user.keytab，且宿主非运行用户。

原因分析

1. 查看日志发现异常信息 Can not found user.keytab in storm.jar。具体信息如下：

```
[main] INFO  b.s.StormSubmitter - Get principle for stream@HADOOP.COM success
[main] ERROR b.s.StormSubmitter - Can not found user.keytab in storm.jar.
Exception in thread "main" java.lang.RuntimeException: Failed to check
principle for keytab
at backtype.storm.StormSubmitter.submitTopologyAs(StormSubmitter.java:219)
at backtype.storm.StormSubmitter.submitTopology(StormSubmitter.java:292)
at backtype.storm.StormSubmitter.submitTopology(StormSubmitter.java:176)
at
com.xxx.streaming.storm.example.hbase.SimpleHBaseTopology.main(SimpleHBaseTopol
ogy.java:77)
```

查看提交的拓扑运行 Jar，发现没有包含 keytab 文件。

2. 查看日志发现异常信息 The submit user is invalid,the principle is 。具体信息如下：

```
[main] INFO  b.s.StormSubmitter - Get principle for stream@HADOOP.COM success
[main] WARN  b.s.s.a.k.ClientCallbackHandler - Could not login: the client is
being asked for a password, but the  client code does not currently support
obtaining a password from the user. Make sure that the client is configured to
use a ticket cache (using the JAAS configuration setting 'useTicketCache=true)'
and restart the client. If you still get this message after that, the TGT in
the ticket cache has expired and must be manually refreshed. To do so, first
determine if you are using a password or a keytab. If the former, run kinit in
a Unix shell in the environment of the user who is running this client using
the command 'kinit <princ>' (where <princ> is the name of the client's Kerberos
principal). If the latter, do 'kinit -k -t <keytab> <princ>' (where <princ> is
the name of the Kerberos principal, and <keytab> is the location of the keytab
file). After manually refreshing your cache, restart this client. If you
continue to see this message after manually refreshing your cache, ensure that
your KDC host's clock is in sync with this host's clock.
[main] ERROR b.s.StormSubmitter - The submit user is invalid,the principle is :
stream@HADOOP.COM
Exception in thread "main" java.lang.RuntimeException: Failed to check
principle for keytab
at backtype.storm.StormSubmitter.submitTopologyAs(StormSubmitter.java:219)
```

```
at backtype.storm.StormSubmitter.submitTopology(StormSubmitter.java:292)
at backtype.storm.StormSubmitter.submitTopology(StormSubmitter.java:176)
at
com.xxx.streaming.storm.example.hbase.SimpleHBaseTopology.main(SimpleHBaseTopol
ogy.java:77)
```

业务提交拓扑时使用的认证用户为 stream，但是在拓扑提交过程中提示 submit user 是无效用户，表明内部校验失败。

3. 查看提交的拓扑运行 Jar，发现包含 keytab 文件。

   查看 user.keytab 文件，发现 principal 为 zmk_kafka.

```
[root@8-5-148-6 client]# klist -kt user.keytab
Keytab name: FILE:user.keytab
KVNO Timestamp         Principal
---- ----------------- --------------------------------------------------------
   1 12/19/16 16:28:17 zmk_kafka@HADOOP.COM
   1 12/19/16 16:28:17 zmk_kafka@HADOOP.COM
```

   发现认证用户和 user.keytab 文件中 principal 不对应。

4. 查看日志发现异常信息 Delete the tmp keytab file failed，the keytab file is ：/tmp/user.keytab，具体信息如下：

```
[main] WARN b.s.StormSubmitter - Delete the tmp keytab file failed, the keytab
file is : /tmp/user.keytab
[main] ERROR b.s.StormSubmitter - The submit user is invalid,the principle is :
hbase1@HADOOP.COM
Exception in thread "main" java.lang.RuntimeException: Failed to check
principle for keytab
at backtype.storm.StormSubmitter.submitTopologyAs(StormSubmitter.java:213)
at backtype.storm.StormSubmitter.submitTopology(StormSubmitter.java:286)
at backtype.storm.StormSubmitter.submitTopology(StormSubmitter.java:170)
at
com.touchstone.storm.cmcc.CmccDataHbaseTopology.main(CmccDataHbaseTopology.java
:183)
```

   查看系统/tmp 目录，发现存在 user.keytab 文件，且文件宿主非运行用户。

解决办法

- 提交拓扑时携带用户 user.keytab 文件。
- 提交拓扑时的用户需要和 user.keytab 文件用户一致。
- 删除/tmp 目录下不对应的 user.keytab 文件。

18.4 提交拓扑后 Worker 日志为空

现象描述

在 Eclipse 中远程提交拓扑成功之后，无法在 Storm WebUI 查看拓扑的详细信息，并且每个拓扑的 Bolt 和 Spout 所在 Worker 节点在一直变化。查看 Worker 日志，日志内容为空。

可能原因

Worker 进程启动失败，触发 Nimbus 重新分配任务，在其他 Supervisor 上启动 Worker。由于 Worker 启动失败后会继续重启，导致 Worker 节点在一直变化，且 Worker 日志内容为空。Worker 进程启动失败的可能原因有两个：

- 提交的 Jar 包中包含“storm.yaml”文件。

  Storm 规定，每个“classpath”中只能包含一个“storm.yaml”文件，如果多于一个那么就会产生异常。使用 Storm 客户端提交拓扑，由于客户端“classpath”配置和 Eclipse 远程提交方式“classpath”不一样，客户端会自动加载用户的 Jar 包到“classpath”，从而使“classpath”中存在两个“storm.yaml”文件。
- Worker 进程初始化时间较长，超过 Storm 集群设置 Worker 启动超时时间，导致 Worker 被 Kill 从而一直进行重分配。

定位思路

1. 使用 Storm 客户端提交拓扑，检查出重复“storm.yaml”问题。
2. 重新打包 Jar 包，然后再提交拓扑。
3. 修改 Storm 集群关于 Worker 启动超时参数。

处理步骤

步骤 1 使用 Eclipse 远程提交拓扑后 Worker 日志为空，则使用 Storm 客户端，提交拓扑对应的 Jar 包，查看提示信息。

例如，Jar 包中包含两个不同路径下的“storm.yaml”文件，系统显示以下信息：

```
Exception in thread "main" java.lang.ExceptionInInitializerError
  at
com.xxx.streaming.storm.example.WordCountTopology.createConf(WordCountTopology.java
:132)
  at
com.xxx.streaming.storm.example.WordCountTopology.remoteSubmit(WordCountTopology.ja
va:120)
  at
com.xxx.streaming.storm.example.WordCountTopology.main(WordCountTopology.java:101)
 Caused by: java.lang.RuntimeException: Found multiple storm.yaml resources. You're
probably bundling the Storm jars with your topology jar.
[jar:file:/opt/xxx/fi_client/Streaming/streaming-
0.9.2/bin/stormDemo.jar!/storm.yaml, file:/opt/xxx/fi_client/Streaming/streaming-
0.9.2/conf/storm.yaml]
  at backtype.storm.utils.Utils.findAndReadConfigFile(Utils.java:151)
  at backtype.storm.utils.Utils.readStormConfig(Utils.java:206)
  at backtype.storm.utils.Utils.<(Utils.java:70)>
```

步骤 2 重新打包 Jar 包，且不能包含“storm.yaml”文件、“log4j”和“slf4j-log4j”相关的 Jar 包。

步骤 3 使用 IntelliJ IDEA 远程提交新打包的 Jar 包。

步骤 4 查看是否可以在 WebUI 查看拓扑的详细信息和 Worker 日志内容。

步骤 5 在 Manager 页面修改 Storm 集群关于 Worker 启动超时参数（参数说明请参考参考信息），保存并重启 Storm 服务。

- MRS Manager 界面操作入口：登录 MRS Manager，依次选择 “服务管理 > Storm> 配置”。
- FusionInsight Manager 界面操作入口：登录 FusionInsight Manager，选择“集群 > 待操作集群的名称 > 服务 > Storm > 配置"

步骤 6 重新提交待运行的 Jar 包。

----结束

参考信息

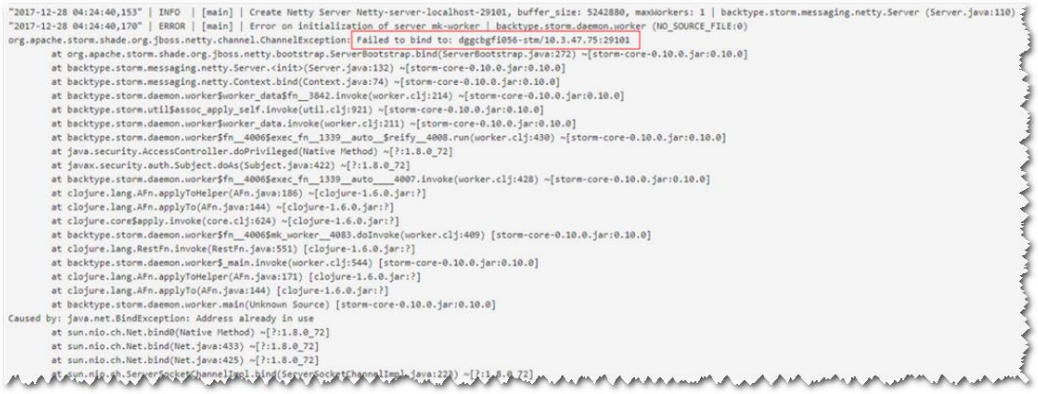
1. nimbus.task.launch.secs 和 supervisor.worker.start.timeout.secs 这两个参数分别代表 nimbus 端和 supervisor 端对于拓扑启动的超时容忍时间，一般 nimbus.task.launch.secs 的值要大于等于 supervisor.worker.start.timeout.secs 的值（建议相等或略大，如果超出太多会影响任务重分配的效率）。
   - nimbus.task.launch.secs：nimbus 在超过该参数配置的时间内没有收到拓扑的 task 发的心跳时，会将该拓扑重新分配（分配给别的 supervisor），同时会刷新 zk 中的任务信息，supervisor 读到 zk 中的任务信息并且与自己当前所启动的拓扑进行比较，如果存在拓扑已经不属于自己，那么则会删除该拓扑的元数据，也就是/srv/Bigdata/streaming_data/stormdir/supervisor/stormdist/{worker-id}目录。
   - supervisor.worker.start.timeout.secs：supervisor 启动 worker 后，在该参数配置的时间内没有收到 worker 的心跳时，supervisor 会主动停掉 worker，等待 worker 的重新调度，一般在业务启动时间较长时适当增加该参数的值，保证 worker 能启动成功。

     如果 supervisor.worker.start.timeout.secs 配置的值比 nimbus.task.launch.secs 的值大，那么则会出现 supervisor 的容忍时间没到，仍然继续让 worker 启动，而 nimbus 却认定该业务启动超时，将该业务分配给了其他主机，这时 supervisor 的后台线程发现任务不一致，删除了拓扑的元数据，导致接下来 worker 在启动过程中要读取 stormconf.ser 时，发现该文件已经不存在了，就会抛出 FileNotFoundException。
2. nimbus.task.timeout.secs 和 supervisor.worker.timeout.secs 这两个参数则分别代表 nimbus 端和 supervisor 端对于拓扑运行过程中心跳上报的超时容忍时间，一般 nimbus.task.timeout.secs 的值要大于等于 supervisor.worker.timeout.secs 的值（建议相等或略大），原理同上。

18.5 提交拓扑后 Worker 运行异常，日志提示 Failed to bind to：host:ip

现象描述

提交业务拓扑后，发现 Worker 无法正常启动。查看 Worker 日志，日志提示 Failed to bind to：host:ip。



可能原因

随机端口范围配置错误。

定位思路

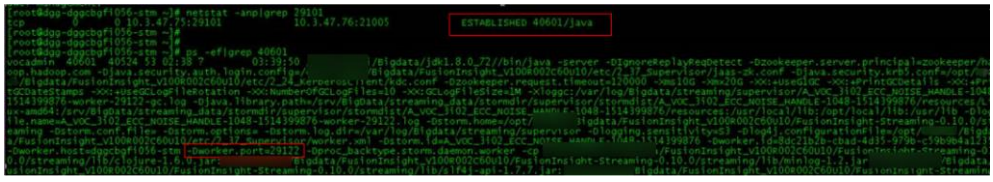
1、检查 worker 相关信息日志。

2、检查绑定端口的进程信息。

3、检查随机端口范围配置。

原因分析

1. 通过 SSH 登录 Worker 启动失败主机，通过 **netstat -anp | grep <port>**命令，查看占用端口的进程 ID 信息。其中 port 修改为实际端口号。
2. 通过 **ps -ef | grep <pid>**命令查看进程的详细信息，其中 pid 为查询出的实际进程 ID。



发现占用端口的进程为 worker 进程，该进程为另一个拓扑业务进程。同时根据进程详细信息发现，分配给该进程的端口为 29122。

3. 通过 **lsof -i:<port>**命令，查看连接详细信息。其中 port 为实际端口号。

发现 29101 端口连接对端端口为 21005，而 21005 为 Kafka 服务端端口。

说明业务层作为客户端连接 Kafka 获取消息，业务端口分配通过 OS 的随机端口分配范围来确定。

4. 通过 **cat /proc/sys/net/ipv4/ip_local_port_range** 命令查看随机端口范围。

5. 发现随机端口范围过大，和 MRS 的服务端口范围存在冲突。

说明

MRS 的服务端口范围：20000-30000。

处理步骤

步骤 1 修改随机端口范围。

```
vi /proc/sys/net/ipv4/ip_local_port_range
32768 61000
```

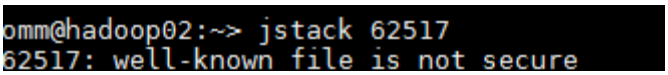
步骤 2 停止占用服务端口的业务进程，释放端口。（停止业务拓扑）

----结束

18.6 使用 jstack 命令查看进程堆栈提示 well-known file is not secure

问题背景与现象

使用 jstack 命令查看进程堆栈信息报：well-known file is not secure。



原因分析

1. 由于执行命令的用户与当前查看 pid 信息的进程提交用户不一致导致。
2. Storm 引入区分用户执行任务特性，在启动 worker 进程时将给进程的 uid 和 gid 改为提交用户和 ficommon，目的是为了 logviewer 可以访问到 worker 进程的日志同时日志文件只开放权限到 640。这样会导致切换到提交用户后对 Worker 进程执行 jstack 和 jmap 等命令执行失败，原因是提交用户的默认 gid 并不是 ficommon，需要通过 ldap 命令修改提交用户的 gid 为 9998（ficommon）才可执行。

解决办法

共有两种方式解决该问题。

方式一：通过 storm 原生页面查看进程堆栈

步骤 1 登录 Storm 原生界面。

MRS Manager 界面操作：

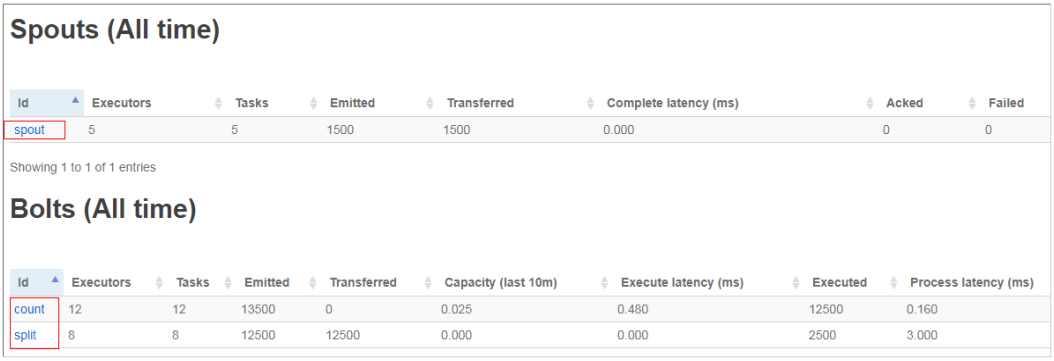
1. 访问 MRS Manager。
2. 在 Manager 选择“服务管理 > Storm”，在“Storm 概述”的“Storm WebUI”，单击任意一个 UI 链接，打开 Storm 的 WebUI。

FusionInsight Manager 界面操作：

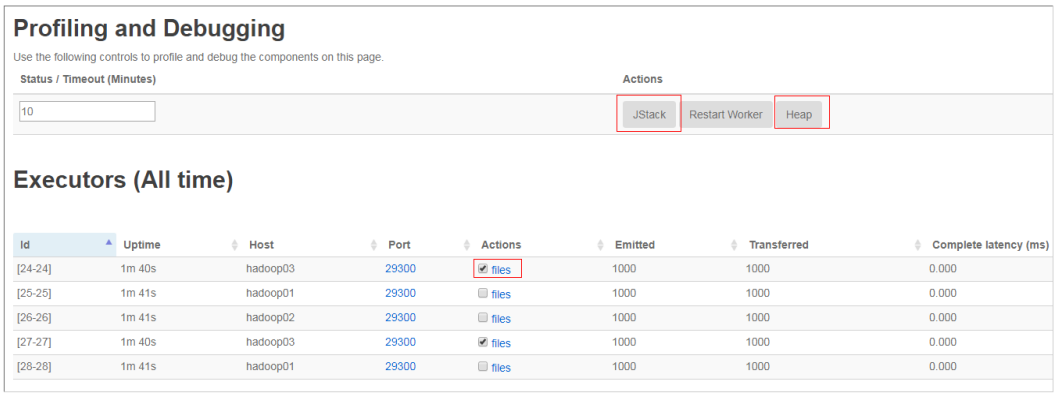
1. 访问 FusionInsight Manager。
2. 在 Manager 选择“集群 > 服务 > Storm”，在“概览”的“Storm WebUI”，单击任意一个 UI 链接，打开 Storm 的 WebUI。

步骤 2 选择要查看的拓扑。

步骤 3 选择要查看的 spout 或者 bolt。



步骤 4 选择要查看的节点日志文件，再选择 JStack 或者 Heap 按钮，其中 JStack 对应的是堆栈信息，Heap 对应的是堆信息：



----结束

方式二：通过修改自定义参数查看进程堆栈

步骤 1 进入 Storm 服务参数配置界面。

MRS Manager 界面操作：登录 MRS Manager 页面，选择“服务管理 > Storm > 服务配置”，“参数类别”选择“全部配置”。

FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > 服务 > Yarn”，单击“配置”，选择“全部配置”。

步骤 2 在左侧导航栏选择“supervisor > 自定义”，添加一个变量 supervisor.run.worker.as.user=false。

步骤 3 保存配置，勾选“重新启动受影响的服务或实例。”并单击“确定”重启服务。

步骤 4 重新提交拓扑。

步骤 5 后台节点切为 omm 用户执行 jps 命令即可查看 worker 的 pid。

```
omm@hadoop02:~> jps | grep worker
22485 worker
111402 worker
```

步骤 6 执行 jstack pid，即可查看 jstack 信息。

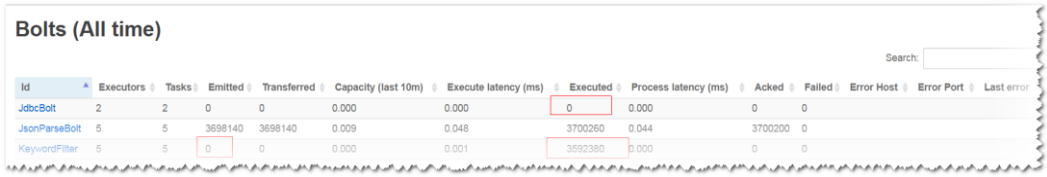
```
omm@hadoop02:~> jstack 22485
2018-05-26 08:46:24
Full thread dump Java HotSpot(TM) 64-Bit Server VM (25.144-b01 mixed mode):

"Attach Listener" #82 daemon prio=9 os_prio=0 tid=0x0000000001c95000 nid=0xb840 waiting on condition [0x0000000000000000]
   java.lang.Thread.State: RUNNABLE

"pool-14-thread-1" #81 daemon prio=5 os_prio=0 tid=0x00007f7ebc931000 nid=0x6113 waiting on condition [0x00007f7eb5ddf000]
   java.lang.Thread.State: TIMED_WAITING (parking)
        at sun.misc.Unsafe.park(Native Method)
        - parking to wait for  <0x00000000dfe820a0> (a java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject)
        at java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:215)
        at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.awaitNanos(AbstractQueuedSynchronizer.java:2078)
        at java.util.concurrent.ScheduledThreadPoolExecutor$DelayedWorkQueue.take(ScheduledThreadPoolExecutor.java:1093)
        at java.util.concurrent.ScheduledThreadPoolExecutor$DelayedWorkQueue.take(ScheduledThreadPoolExecutor.java:809)
        at java.util.concurrent.ThreadPoolExecutor.getTask(ThreadPoolExecutor.java:1074)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1134)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
        at java.lang.Thread.run(Thread.java:748)
```

----结束

18.7 使用 Storm-JDBC 插件开发 Oracle 写入 Bolt，发现数据无法写入

现象描述

使用 Storm-JDBC 插件开发 Oracle 写入 Bolt，发现能连上 Oracle 数据库，但是无法向 Oracle 数据库里面写数据。

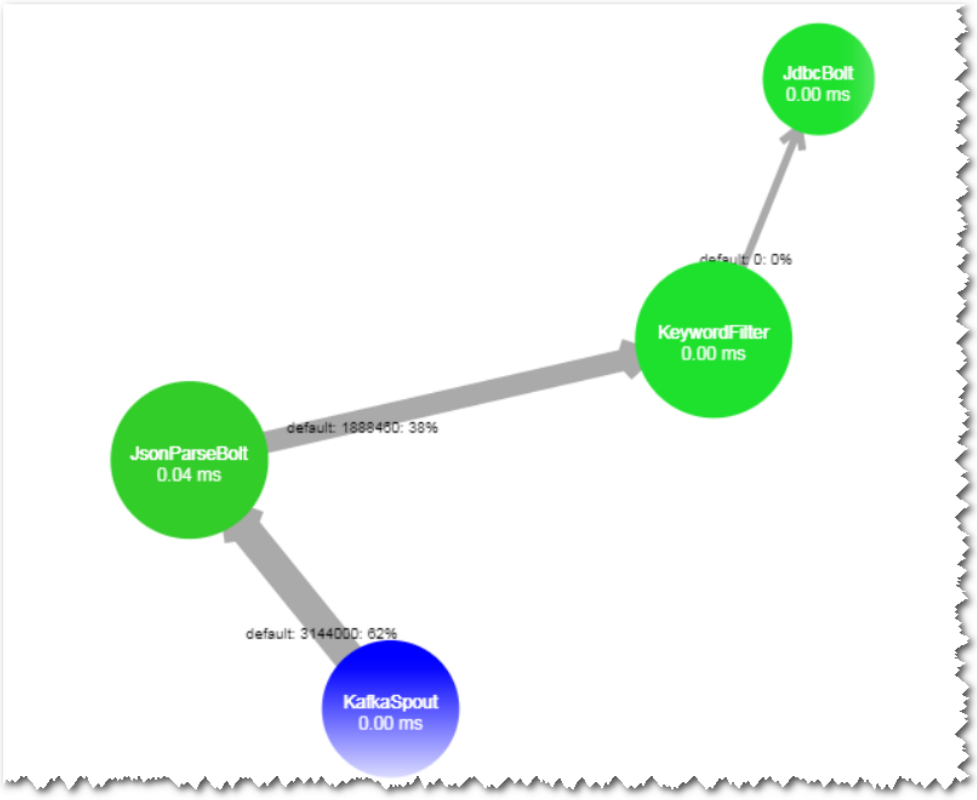


可能原因

- 拓扑定义异常。
- 数据库表结果定义异常。

原因分析

1. 通过 Storm WebUI 查看拓扑 DAG 图，发现 DAG 图与拓扑定义一致。

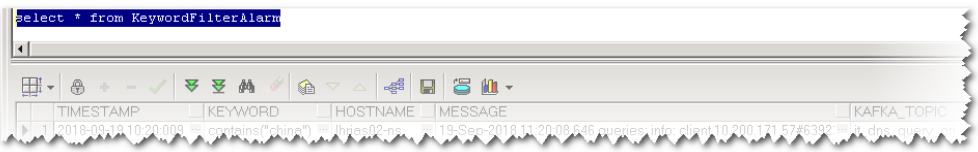


2. 查看 KeyWordFilter Bolt 输出流字段定义和发送消息字段发现一致。

```
@Override
public void declareOutputFields(OutputFieldsDeclarer declarer)
{
    declarer.declare(new Fields("timestamp","keyword","hostname","message","kafka_topic") );
```

```
    if( flag )
    {
        String keyword = expParser.getKeyword();
        System.out.println( message );
        collector.emit(new Values( timestamp, keyword , hostname , message, kafka_topic ));
```

3. 查看 Oracle 数据库中表定义，发现字段名为大写，与流定义字段名称不一致。



4. 单独调试 execute 方法，发现抛出字段不存在。

处理步骤

修改流定义字段名称为大写，与数据库表定义字段一致。

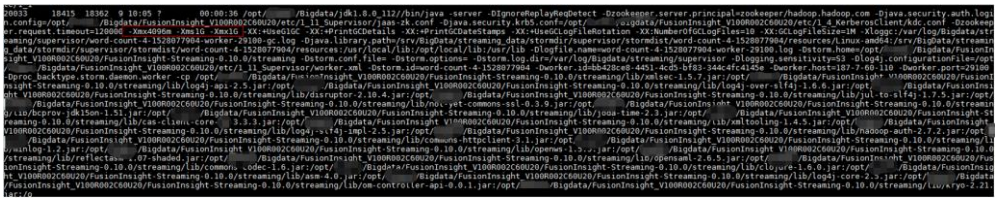
18.8 业务拓扑配置 GC 参数不生效

问题背景与现象

业务拓扑代码中配置参数 topology.worker.childopts 不生效，关键日志如下：

```
[main] INFO b.s.StormSubmitter - Uploading topology jar /opt/jar/example.jar to
assigned location: /srv/BigData/streaming/stormdir/nimbus/inbox/stormjar-8d3b778d-
69ea-4fbe-ba88-01aa2036d753.jar
Start uploading file '/opt/jar/example.jar' to
'/srv/BigData/streaming/stormdir/nimbus/inbox/stormjar-8d3b778d-69ea-4fbe-ba88-
01aa2036d753.jar' (65574612 bytes)
[==================================================] 65574612 / 65574612
File '/opt/jar/example.jar' uploaded to
'/srv/BigData/streaming/stormdir/nimbus/inbox/stormjar-8d3b778d-69ea-4fbe-ba88-
01aa2036d753.jar' (65574612 bytes)
[main] INFO b.s.StormSubmitter - Successfully uploaded topology jar to assigned
location: /srv/BigData/streaming/stormdir/nimbus/inbox/stormjar-8d3b778d-69ea-4fbe-
ba88-01aa2036d753.jar
[main] INFO b.s.StormSubmitter - Submitting topology word-count in distributed mode
with conf {"topology.worker.childopts":"-
Xmx4096m","storm.zookeeper.topology.auth.scheme":"digest","storm.zookeeper.topology
.auth.payload":"-5915065013522446406:-6421330379815193999","topology.workers":1}
[main] INFO b.s.StormSubmitter - Finished submitting topology: word-count
```

通过 ps -ef | grep worker 命令查看 worker 进程信息如下：



原因分析

1. 由于 topology.worker.gc.childopts、topology.worker.childopts 和 worker.gc.childopts(服务端参数)有优先级，优先级大小为：topology.worker.gc.childopts > worker.gc.childopts > topology.worker.childopts。

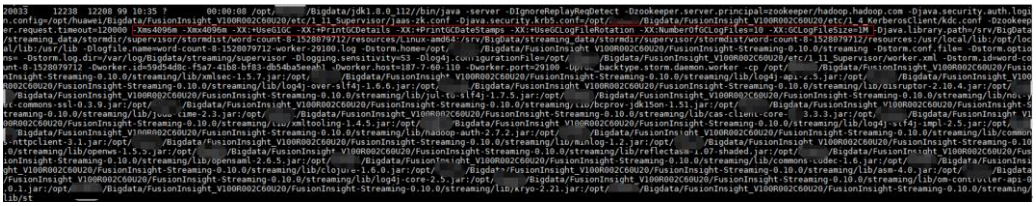
2. 如果设置了客户端参数 topology.worker.childopts，则该参数会与服务端参数 worker.gc.childopts 共同配置，但是后面的相同参数会将前面的覆盖掉，如上面图有两个-Xmx，-Xmx1G 会覆盖掉-Xmx4096m。
3. 如果配置了 topology.worker.gc.childopts 则服务端参数 worker.gc.childopts 会被替换。

解决办法

步骤 1 如果想要修改拓扑的 JVM 参数，可以在命令中直接修改 topology.worker.gc.childopts 这个参数或者在服务端修改该参数，当 topology.worker.gc.childopts 为 "-Xms4096m -Xmx4096m -XX:+UseG1GC -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -XX:GCLogFileSize=1M"时，效果如下：

```
[main-SendThread(10.7.61.88:2181)] INFO o.a.s.s.o.a.z.ClientCnxn - Socket
connection established, initiating session, client: /10.7.61.88:44694, server:
10.7.61.88/10.7.61.88:2181
[main-SendThread(10.7.61.88:2181)] INFO o.a.s.s.o.a.z.ClientCnxn - Session
establishment complete on server 10.7.61.88/10.7.61.88:2181, sessionid =
0x16037a6e5f092575, negotiated timeout = 40000
[main-EventThread] INFO o.a.s.s.o.a.c.f.s.ConnectionStateManager - State change:
CONNECTED
[main] INFO b.s.u.StormBoundedExponentialBackoffRetry - The baseSleepTimeMs [1000]
the maxSleepTimeMs [1000] the maxRetries [1]
[main] INFO o.a.s.s.o.a.z.Login - successfully logged in.
[main-EventThread] INFO o.a.s.s.o.a.z.ClientCnxn - EventThread shut down for
session: 0x16037a6e5f092575
[main] INFO o.a.s.s.o.a.z.ZooKeeper - Session: 0x16037a6e5f092575 closed
[main] INFO b.s.StormSubmitter - Uploading topology jar /opt/jar/example.jar to
assigned location: /srv/BigData/streaming/stormdir/nimbus/inbox/stormjar-86855b6b-
133e-478d-b415-fa96e63e553f.jar
Start uploading file '/opt/jar/example.jar' to
'/srv/BigData/streaming/stormdir/nimbus/inbox/stormjar-86855b6b-133e-478d-b415-
fa96e63e553f.jar' (74143745 bytes)
[==================================================] 74143745 / 74143745
File '/opt/jar/example.jar' uploaded to
'/srv/BigData/streaming/stormdir/nimbus/inbox/stormjar-86855b6b-133e-478d-b415-
fa96e63e553f.jar' (74143745 bytes)
[main] INFO b.s.StormSubmitter - Successfully uploaded topology jar to assigned
location: /srv/BigData/streaming/stormdir/nimbus/inbox/stormjar-86855b6b-133e-478d-
b415-fa96e63e553f.jar
[main] INFO b.s.StormSubmitter - Submitting topology word-count in distributed mode
with conf
{"storm.zookeeper.topology.auth.scheme":"digest","storm.zookeeper.topology.auth.pay
load":"-7360002804241426074:-6868950379453400421","topology.worker.gc.childopts":"-
Xms4096m -Xmx4096m -XX:+UseG1GC -XX:+PrintGCDetails -XX:+PrintGCDateStamps -
XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -
XX:GCLogFileSize=1M","topology.workers":1}
[main] INFO b.s.StormSubmitter - Finished submitting topology: word-count
```

步骤 2 通过 ps -ef | grep worker 命令查看 worker 进程信息如下：



----结束

18.9 UI 查看信息显示 Internal Server Error

问题背景与现象

使用 MRS 版本安装集群，主要安装 ZooKeeper、Strom。

通过 MRS Manager 中的 Storm Status 页面 UI 连接访问信息时出现 Internal Server Error。

UI 页面出现如下信息：

```
Internal Server Error
org.apache.thrift7.transport.TTransportException: Frame size (306030) larger than
max length (1048576)!
```

可能原因

- Storm 服务中 Nimbus 异常。
- Storm 集群信息较多超过系统默认 Thrift 传输大小的设置。

原因分析

1. 查看 Storm 服务状态及监控指标：
   - MRS Manager 界面操作：登录 MRS Manager，依次选择 "服务管理 > Storm ，查看当前 Storm 状态，发现状态为良好，且监控指标内容显示正确。
   - FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > 待操作集群的名称 > 服务 > Storm，

     查看当前 Storm 状态，发现状态为良好，且监控指标内容显示正确。
2. 选择“实例”页签，查看 Nimbus 实例状态，显示正常。
3. 查看当前 Storm 集群 thrift 相关配置，发现 nimbus.thrift.max_buffer_size 参数配置为 1048576（1M）。
4. 上述配置和异常信息中信息一致，说明当前配置的 Thrift 的 buffer size 小于集群信息所需的 buffer size。

解决方法

调整 Storm 集群中 Thrift 的 Buffer Size 大小，具体大小根据错误信息进行实际调整。

步骤 1 进入 Storm 服务参数配置界面。

- MRS Manager 界面操作：登录 MRS Manager 页面，选择“服务管理 > Storm > 服务配置”，“参数类别”选择“全部配置”。
- FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > 服务 > Yarn”，单击“配置”，选择“全部配置”。

步骤 2 修改 nimbus.thrift.max_buffer_size 参数为 10485760（10M）。

步骤 3 保存配置，勾选“重新启动受影响的服务或实例。”并单击“确定”重启服务。

----结束

19 使用 Ranger

19.1 Hive 启用 Ranger 鉴权后，在 Hue 页面能查看到没有权限的表和库

用户问题

Hive 启用 Ranger 鉴权后，在 Hue 页面能查看到没有权限的表和库

问题现象

普通集群（未开启 Kerberos 认证）中，Hive 启用 Ranger 鉴权后，在 Hue 页面能查看到没有权限的表和库。

原因分析

Hive 启用 Ranger 鉴权后，默认的 Hive 策略中有 2 个关于 database 的 public 组策略，所有用户都属于 public 组，默认给 public 组配有 default 数据库的创表和所有其他数据库的 **create** 权限，因此默认所有的用户都有 **show databases** 和 **show tables** 的权限，如果不想让某些用户有 **show databases** 和 **show tables** 权限，可在 Ranger WEBUI 中删除该默认 public 组策略，并赋予需要查看的用户权限，具体请参考处理步骤。

处理步骤

步骤 1 登录 Ranger WebUI 界面。

步骤 2 在“Service Manager”区域内，单击 Hive 组件名称，进入 Hive 组件安全访问策略列表页面。

步骤 3 分别单击“all - database”和“default database tables columns”策略所在行的按钮。

步骤 4 删除“public”组策略。

图19-1 all - database 策略

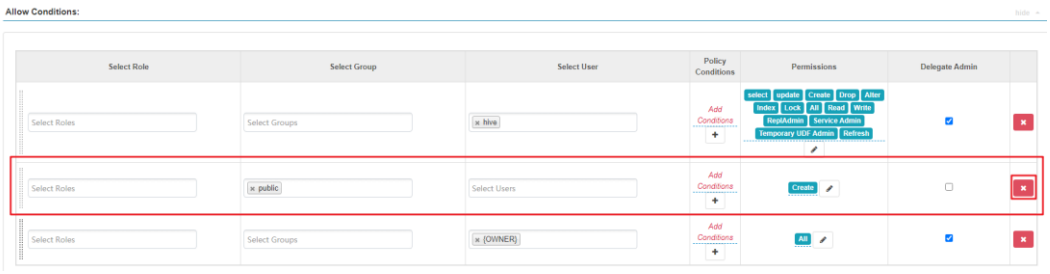
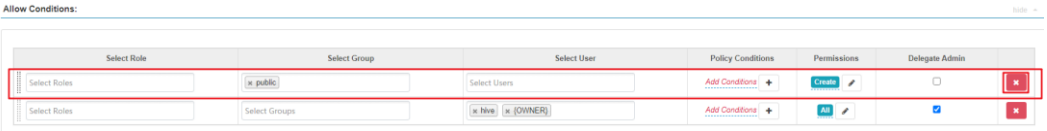


图19-2 default database tables columns 策略



步骤 5 在 Hive 组件安全访问策略列表页面，单击“Add New Policy”为相关用户或者用户组添加资源访问策略。

----结束

20 使用 Yarn

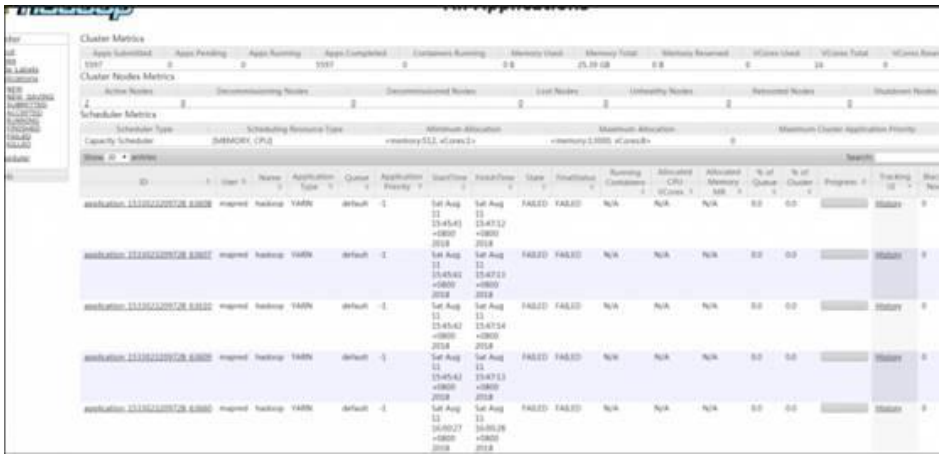
20.1 启动 Yarn 后发现一堆 job

用户问题

MRS 2.x 及之前版本集群，构建 MRS 集群启动 Yarn 后，发现一堆 job 占用资源。

问题现象

客户使用 MapReduce 服务构建集群启动 Yarn 后 生成一堆 job 占用资源。



原因分析

- 疑似黑客攻击。
- 安全组入口方向的 Any 协议源地址配置为 0.0.0.0/0。

IPv4	Any	Any	0.0.0.0/0
IPv4	Any	Any	0.0.0.0/0
IPv4	Any	Any	0.0.0.0/0

处理步骤

步骤 1 登录 MRS 集群页面，在“现有集群”中，单击对应的集群名称，进入集群详情页面。

步骤 2 单击“集群管理页面”后面的“前往 Manager”，弹出“访问 MRS Manager 页面”。

步骤 3 单击“管理安全组规则”，检查安全组规则配置。

步骤 4 检查入口方向 Any 协议的源地址是否为 0.0.0.0/0。

步骤 5 如果是，修改入口方向 Any 协议的远端为指定 IP 地址。如果不是，则无需修改。

步骤 6 修改成功后，重启集群虚机。

----结束

建议与总结

关闭入口方向的 Any 协议，或者指定入口方向的 Any 协议远端为指定 IP。

参考信息

请参考。

20.2 通过客户端 hadoop jar 命令提交任务，客户端返回 GC overhead

问题背景与现象

通过客户端提交任务，客户端返回内存溢出的报错结果:

```
main path:hdfs://hacluster/user/wangyou
17/09/18 08:29:57 INFO hdfs.DFSClient: Created HDFS_DELEGATION_TOKEN token 22890097 for wangyou on ha-hdfs:hacluster
17/09/18 08:29:57 INFO security.TokenCache: Got dt for hdfs://hacluster; Kind: HDFS_DELEGATION_TOKEN, Service: ha-hdfs:hacluster, Ident: (HDFS_DELEGATION_TOKEN token 2
2890097 for wangyou)
17/09/18 08:29:57 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with
ToolRunner to remedy this.
17/09/18 08:32:42 INFO retry.RetryInvocationHandler: Exception while invoking getListing of class ClientNamenodeProtocolTranslatorPB over fi1-cn-003/10.113.246.10:2500
0. Trying to fail over immediately.
java.io.IOException: com.google.protobuf.ServiceException: java.lang.OutOfMemoryError: GC overhead limit exceeded
        at org.apache.hadoop.ipc.ProtobufHelper.getRemoteException(ProtobufHelper.java:47)
        at org.apache.hadoop.hdfs.protocolPB.ClientNamenodeProtocolTranslatorPB.getListing(ClientNamenodeProtocolTranslatorPB.java:578)
        at sun.reflect.GeneratedMethodAccessor2.invoke(Unknown Source)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:497)
        at org.apache.hadoop.io.retry.RetryInvocationHandler.invokeMethod(RetryInvocationHandler.java:191)
        at org.apache.hadoop.io.retry.RetryInvocationHandler.invoke(RetryInvocationHandler.java:102)
        at com.sun.proxy.$Proxy10.getListing(Unknown Source)
        at org.apache.hadoop.hdfs.DFSClient.listPaths(DFSClient.java:1757)
        at org.apache.hadoop.hdfs.DistributedFileSystem$DirListingIterator.hasNextNoFilter(DistributedFileSystem.java:1024)
        at org.apache.hadoop.hdfs.DistributedFileSystem$DirListingIterator.hasNext(DistributedFileSystem.java:999)
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.singleThreadedListStatus(FileInputFormat.java:304)
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.listStatus(FileInputFormat.java:265)
        at org.apache.hadoop.mapreduce.lib.input.CombineFileInputFormat.getSplits(CombineFileInputFormat.java:217)
        at org.apache.hadoop.mapreduce.lib.input.DelegatingInputFormat.getSplits(DelegatingInputFormat.java:115)
        at org.apache.hadoop.mapreduce.JobSubmitter.writeNewSplits(JobSubmitter.java:306)
        at org.apache.hadoop.mapreduce.JobSubmitter.writeSplits(JobSubmitter.java:323)
        at org.apache.hadoop.mapreduce.JobSubmitter.submitJobInternal(JobSubmitter.java:200)
        at org.apache.hadoop.mapreduce.Job$10.run(Job.java:1290)
        at org.apache.hadoop.mapreduce.Job$10.run(Job.java:1287)
        at java.security.AccessController.doPrivileged(Native Method)
        at javax.security.auth.Subject.doAs(Subject.java:422)
        at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1673)
        at org.apache.hadoop.mapreduce.Job.submit(Job.java:1287)
```

原因分析

从报错堆栈可以看出是任务在提交过程中分片时在读取 HDFS 文件阶段内存溢出了，一般是由于该任务要读取的小文件很多导致内存不足。

解决办法

步骤 1 排查启动的 MR 任务是否对应的 HDFS 文件个数很多，如果很多，减少文件数量，提前先合并小文件或者尝试使用 combineInputFormat 来减少任务读取的文件数量。

步骤 2 增大 hadoop 命令执行时的内存，该内存在客户端中设置，修改对应路径“*客户端安装目录*/HDFS/component_env”文件中“CLIENT_GC_OPTS”的“-Xmx”参数，将该参数的默认值改大，比如改为 512m。然后执行 **source component_env** 命令，使修改的参数生效。

```
export YARN_ROOT_LOGGER=INFO,console

#GC_OPTS for client operation.
CLIENT_GC_OPTS="-Xmx512m -Djava.io.tmpdir=${HADOOP_HOME}"

export HADOOP_CLIENT_OPTS="$CLIENT_GC_OPTS"
```

----结束

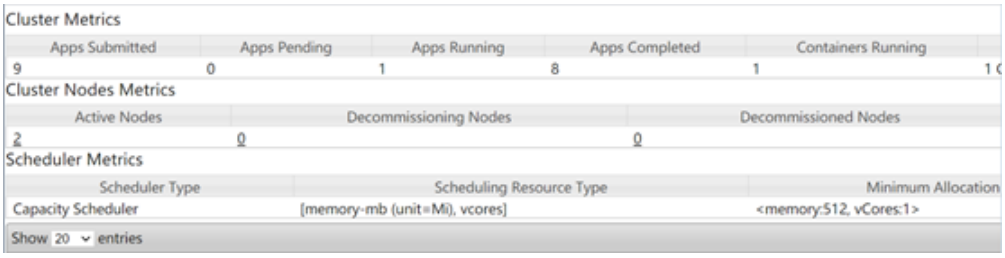
20.3 Yarn 汇聚日志过大导致磁盘被占满

用户问题

集群的磁盘使用率很高。

问题现象

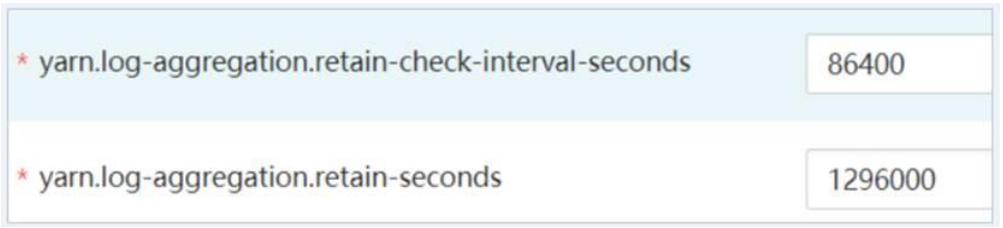
- Manager 管理页面下主机管理显示磁盘使用率过高。
- Yarn WebUI 界面上显示只有少量任务在运行。



- 登录到集群的 Master 节点执行 **hdfs dfs -du -h /** 命令发现如下文件占用大量磁盘空间。

```
22.5 G   45.0 G   /tmp/logs/root/logs/application_1589278244866_0153
18.4 M   36.8 M   /tmp/logs/root/logs/application_1589278244866_0154
23.4 G   46.8 G   /tmp/logs/root/logs/application_1589278244866_0155
23.5 G   46.9 G   /tmp/logs/root/logs/application_1589278244866_0156
23.7 G   47.4 G   /tmp/logs/root/logs/application_1589278244866_0157
23.7 G   47.4 G   /tmp/logs/root/logs/application_1589278244866_0158
22.5 G   45.0 G   /tmp/logs/root/logs/application_1589278244866_0159
18.5 M   37.0 M   /tmp/logs/root/logs/application_1589278244866_0160
22.5 G   45.0 G   /tmp/logs/root/logs/application_1589278244866_0161
18.8 M   37.6 M   /tmp/logs/root/logs/application_1589278244866_0162
24.0 G   48.0 G   /tmp/logs/root/logs/application_1589278244866_0163
121.3 K  242.7 K  /tmp/logs/root/logs/application_1589278244866_0164
1.1 M    2.1 M    /tmp/logs/root/logs/application_1589278244866_0165
1.1 M    2.1 M    /tmp/logs/root/logs/application_1589278244866_0166
1.1 M    2.1 M    /tmp/logs/root/logs/application_1589278244866_0167
1.1 M    2.1 M    /tmp/logs/root/logs/application_1589278244866_0168
```

- Yarn 服务的汇聚日志配置如下



原因分析

客户提交任务的操作过于频繁，且聚合后的日志文件被删除的时间配置为 1296000，即聚合日志保留 15 天，导致汇聚的日志无法在短时时间内释放，从而引起磁盘被占满。

处理步骤

步骤 1 登录 Manager 页面，进入 MapReduce 服务全部配置页面。

- MRS Manager 界面操作：登录 MRS Manager，选择“服务管理 > MapReduce > 服务配置 > 全部配置”。
- FusionInsight Manager 界面操作：登录 FusionInsight Manager，选择“集群 > 服务 > MapReduce > 配置 > 全部配置”。

步骤 2 搜索“yarn.log-aggregation.retain-seconds”参数，并根据实际情况将 yarn.log-aggregation.retain-seconds 调小，比如调整为： 259200，即 Yarn 的聚合日志保留 3 天，到期后自动释放磁盘空间。

步骤 3 保存配置，不勾选“重新启动受影响的服务或实例”。

步骤 4 在业务空闲时执行该步骤重启 MapReduce 服务，重启服务会导致上层服务业务中断，影响集群的管理维护和业务，建议在空闲时执行。

1. 登录 Manager 页面。
2. 重启 MapReduce 服务。

----结束

20.4 MR 任务异常临时文件不删除

用户问题

MR 任务异常临时文件为什么没有删除？

问题现象

HDFS 临时目录文件过多，占用内存。

原因分析

MR 任务提交时会将相关配置文件、jar 包和-files 添加的文件都放入 hdfs 上的临时目录，方便后面 container 启动以后获取相应的文件。由配置项 yarn.app.mapreduce.am.staging-dir 决定具体存放位置，默认值是/tmp/hadoop-yarn/staging。

正常运行的 MR 任务会在 Job 结束以后就清理这些临时文件，但是当 Job 对应的 yarn 任务是异常退出时，这些临时文件不会被清理，长时间积攒导致该临时目录下的文件数量越来越多，占用存储空间越来越多。

处理步骤

步骤 1 登录集群。

1. 以 root 用户登录任意一个 Master 节点，用户密码为创建集群时用户自定义的密码。
2. 如果集群开启 Kerberos 认证，执行如下命令进入客户端安装目录并设置环境变量，再认证用户并按照提示输入密码，该密码请向管理员获取。

   cd ***客户端安装目录***

   source bigdata_env

   kinit hdfs
3. 如果集群未开启 Kerberos 认证，执行如下命令切换到 omm 用户，再进入客户端安装目录设置环境变量。

   su - omm

   cd ***客户端安装目录***

   source bigdata_env

步骤 2 获取文件列表。

hdfs dfs -ls /tmp/hadoop-yarn/staging/*/.staging/ | grep "^drwx" | awk '{print $8}' > job_file_list

job_file_list 文件中就是所有任务的文件夹列表，文件内容参考：

```
/tmp/hadoop-yarn/staging/omm/.staging/job__<Timestamp>_<ID>
```

步骤 3 统计当前运行中的任务。

mapred job -list 2>/dev/null | grep job_ | awk '{print $1}' > run_job_list

run_job_list 文件里面就是当前正在运行的 JobId 列表，文件内容格式为：

```
job_<Timestamp>_<ID>
```

步骤 4 删除 job_file_list 文件中正在运行中的任务。确保在删除过期数据时不会误删正在运行任务的数据。

cat run_job_list | while read line; do sed -i "/$line/d" job_file_list; done

步骤 5 删除过期数据。

cat job_file_list | while read line; do hdfs dfs -rm -r $line; done

步骤 6 清除临时文件。

rm -rf run_job_list job_file_list

----结束

20.5 提交任务的 Yarn 的 ResourceManager 报错 connection refused，且配置的 Yarn 端口为 8032

用户问题

请求提交任务的 Yarn 的 ResourceManager 报错 connection refused，且配置的 Yarn 端口为 8032。

问题现象

MRS 的 Yarn ResourceManager 中的一个节点无法连接，且配置的 Yarn 端口为 8032。

原因分析

该业务应用在集群外部运行，且使用的是老集群的客户端，配的 Yarn 端口是 8032，与 MRS 服务的 Yarn ResourceManager 实际端口不同。从而使请求提交任务的 Yarn 的 ResourceManager 报错 connection refused。

处理步骤

步骤 1 更新 MRS 服务客户端。

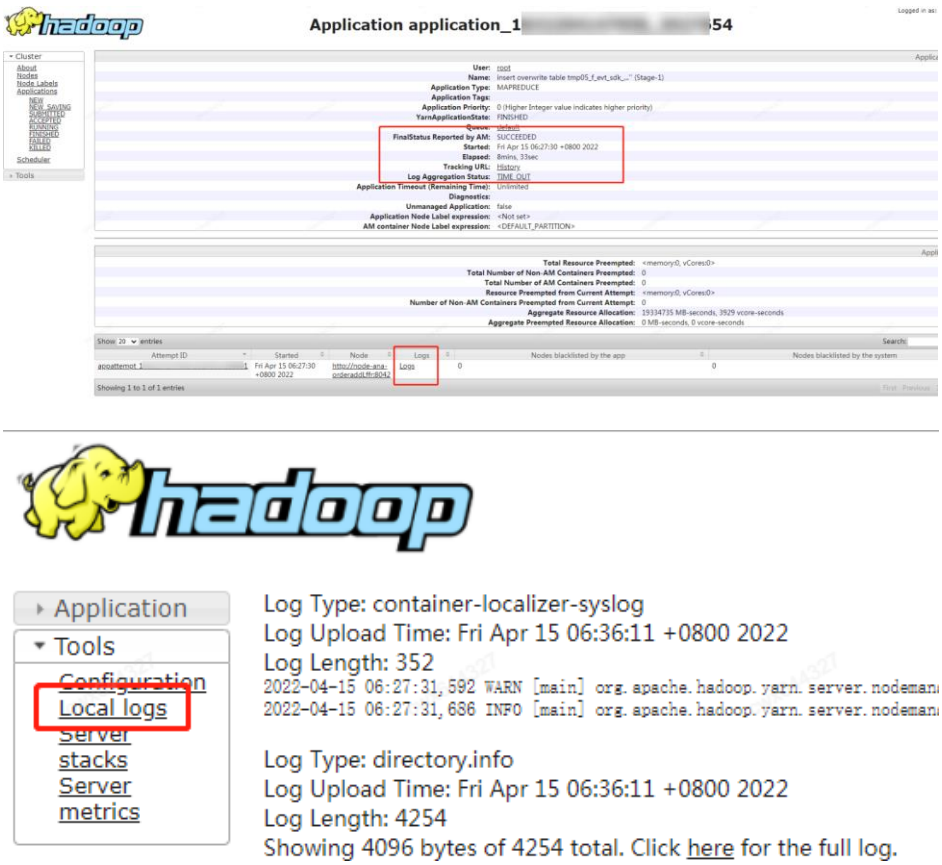
步骤 2 然后重试提交作业。

----结束

20.6 Yarn WebUI 作业查看日志提示“Could not access logs page!”

问题背景与现象

登录 Yarn WebUI 界面查看作业日志“Logs”，然后单击“Local logs”，界面提示“Could not access logs page!”



原因分析

该 Local logs 是用来访问服务的日志，但由于安全考虑，该功能暂不支持从 Yarn WebUI 界面访问。您可以登录 ResourceManager 主节点查看 ResourceManager 的日志。

处理步骤

步骤 1 登录 Manager 界面，选择“集群 > 组件 > Yarn > 实例”，查看并获取“ResourceManager(主)”实例的业务 IP 地址。

步骤 2 以 **root** 用户登录 ResourceManager 主节点。

步骤 3 进入“/var/log/Bigdata/yarn/rm”路径查看 ResourceManager 的日志。

cd /var/log/Bigdata/yarn/rm

----结束

20.7 Yarn 页面单击队列名称报错

问题背景与现象

在 Yarn 使用 Capacity 调度器时，单击 Yarn 原生页面的队列名称会报 500 的错误。

HTTP ERROR 500 javax.servlet.ServletException: javax.servlet.ServletException: java.lang.IllegalArgumentException: Illegal character in query at index 81: https://[redacted]:20026/Yarn/ResourceManager/21/cluster/scheduler?openQueues=^default$

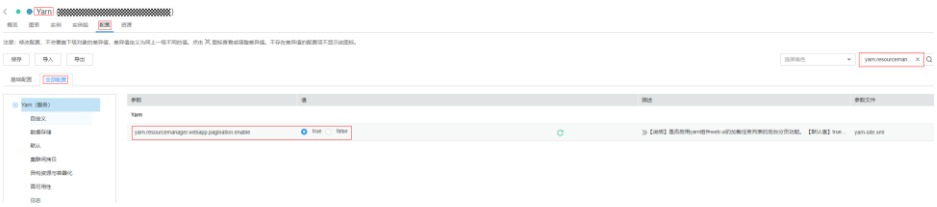
原因分析

页面链接无法识别符号“^”，导致页面访问失败。

处理步骤

步骤 1 登录 Manager 页面，选择“集群 > 服务 > Yarn > 配置 > 全部配置”。

步骤 2 在搜索框搜索“yarn.resourcemanager.webapp.pagination.enable”。



步骤 3 如果该参数值为“true”（默认为“true”），请修改为“false”，并保存配置。

步骤 4 在 Yarn 服务页面选择“实例”页签，勾选所有的 ResourceManager 实例，选择“更多 > 滚动重启实例”，等待实例启动完成。

----结束

20.8 TimelineServer 目录文件数量到达上限

问题现象

MRS 3.x 版本集群，ResourceManager 日志显示 TimelineServer 数据目录数量到达上限打印大量错误日志。

异常日志内容如下：

```
The directory item limit of /tmp/hadoop-omm/yarn/timeline/generic-history/ApplicationHistoryDataRoot is exceeded: limit=1048576 items=1048576
```

原因分析

TimelineServer 在 MRS 3.x 版本会使用一个 HDFS 的目录（例如以上报错中的“/tmp/hadoop-omm/yarn/timeline/generic-history/ApplicationHistoryDataRoot”路径）来存放历史任务信息，导致该目录下的文件不断累积，直到到达 HDFS 配置的目录数量上限（“dfs.namenode.fs-limits.max-directory-items”默认为“1048576”）。

此时请将“yarn.timeline-service.generic-application-history.enabled”（客户端查询 app 任务数据时是否从 TimelineServer 服务获取）参数设置为“false”，直接从 ResourceManager 上面获取 app 任务数据。

处理步骤

步骤 1 登录 FusionInsight Manager，选择“集群 > 服务 > Yarn > 配置 > 全部配置”。

步骤 2 在左侧导航栏选择“Yarn（服务） > 自定义”，在自定义页面的“yarn.yarn-site.customized.configs”参数后添加“yarn.timeline-service.generic-application-history.enabled”，值为“false”，单击“保存”。

步骤 3 滚动重启 ResourceManager 和 TimelineServer 实例。

单击 Yarn 服务的“实例”页签，勾选所有 ResourceManager 和 TimelineServer 实例，选择“更多 > 滚动重启实例”。

步骤 4 （可选）根据业务需要，在业务低峰时滚动重启 NodeManager。

步骤 5 实例重启完成后，删除 HDFS 上的存放历史任务信息的目录，例如“/tmp/hadoop-omm/yarn/timeline/generic-history/ApplicationHistoryDataRoot”。

1. 以客户端安装用户登录客户端安装目录，并配置环境变量。

   cd *客户端安装目录*

   source bigdata_env

2. 执行以下命令认证用户，未开启 Kerberos 认证的用户跳过该步骤。

   kinit *业务用户*

3. 执行以下命令删除 HDFS 上的相关目录。

   hdfs dfs -rm -r */tmp/hadoop-omm/yarn/timeline/generic-history/ApplicationHistoryDataRoot/*

----结束

21 使用 ZooKeeper

21.1 MRS 集群如何访问 ZooKeeper

用户问题

MRS 集群如何访问 ZooKeeper?

问题现象

客户在 MRS 的 Master 节点使用 zkcli.sh 访问 ZooKeeper 但是存在报错。

原因分析

客户使用命令有问题，造成报错的发生。

处理步骤

步骤 1 获取 ZooKeeper 的 IP 地址。

步骤 2 以 root 用户登录 Master 节点。

步骤 3 初始化环境变量。

source /opt/client/bigdata_env

步骤 4 执行 **zkCli.sh -server** *ZooKeeper 所在节点的 IP*:**2181** 即可连接上 MRS 的 ZooKeeper。

zk 所在节点的 IP 即为步骤 1 中查到的结果，多个 IP 之间以逗号间隔。

步骤 5 使用 **ls /**等常用的命令查看 ZooKeeper 上的信息。

----结束

22 访问 OBS

22.1 使用 MRS 多用户访问 OBS 功能时/tmp 目录没有权限

用户问题

在使用 MRS 多用户访问 OBS 功能，执行 spark、hive、presto 等作业时，出现/tmp 目录没有权限的报错。

问题现象

在使用 MRS 多用户访问 OBS 功能，执行 spark、hive、presto 等作业时，出现/tmp 目录没有权限的报错。

原因分析

作业执行过程中有临时目录，提交作业的用户对临时目录没有权限。

处理步骤

步骤 1 在集群“概览”页签中，查询并记录集群所绑定的委托名称。

步骤 2 登录 IAM 服务控制台。

步骤 3 选择“权限 > 创建自定义策略”。

- 策略名称：请输入策略名称。
- 作用范围：请选择“全局级服务”。
- 策略配置方式：请选择“可视化视图”。
- 策略内容：
  a. “允许”选择“允许”。
  b. “云服务”选择“对象存储服务 (OBS)”。
  c. “操作”勾选所有“写”、“列表”和“只读”权限。
  d. “特定资源”选择：

i. “object”选择“通过资源路径指定”，并单击“添加资源路径”分别在“路径”中输入 ***obs_bucket_name*/tmp/**和 ***obs_bucket_name*/tmp/***。此处以**/tmp** 目录为例，如需其他目录权限请参考该步骤添加对应目录及该目录下所有对象的资源路径。

ii. “bucket”选择“通过资源路径指定”，并单击“添加资源路径”在“路径”中输入 ***obs_bucket_name***。

其中 **obs_bucket-name** 请使用实际的 OBS 桶名替换。若桶类型为“并行文件系统”需要在添加 ***obs_bucket_name*/tmp/**路径，桶类型为“对象存储”则不需要添加 ***obs_bucket_name*/tmp/**路径。

e. （可选）请求条件，暂不添加。

图22-1 自定义策略

步骤 4 单击“确定”完成策略添加。

步骤 5 选择“委托”，并在步骤 1 中查询到的委托所在行的“操作”列单击“权限配置”。

步骤 6 查询并勾选步骤 3 中创建的策略。

步骤 7 单击“确定”完成委托权限配置。

----结束

22.2 Hadoop 客户端删除 OBS 上数据时.Trash 目录没有权限

用户问题

使用 Hadoop 客户端删除 OBS 上数据时出现.Trash 目录没有权限的报错。

问题现象

执行 **hadoop fs -rm obs://<obs_path>**出现如下报错：

```
exception [java.nio.file.AccessDeniedException: user/root/.Trash/Current/:
getFileStatus on user/root/.Trash/Current/: status [403]
```

原因分析

hadoop 删除文件时会将文件先移动到.Trash 目录，若该目录没有权限则出现 403 报错。

处理步骤

方案一：

使用 **hadoop fs -rm -skipTrash** 来删除文件。

方案二：

在集群对应的委托中添加访问.Trash 目录的权限。

步骤 1 在集群“概览”页签中，查询并记录集群所绑定的委托名称。

步骤 2 登录 IAM 服务控制台。

步骤 3 选择“权限 > 创建自定义策略”。

- 策略名称：请输入策略名称。
- 作用范围：请选择“全局级服务”。
- 策略配置方式：请选择“可视化视图”。
- 策略内容：
  a. “允许”选择“允许”。
  b. “云服务”选择“对象存储服务 (OBS)”。
  c. “操作”勾选所有操作权限。
  d. “特定资源”选择：
     i. “object”选择“通过资源路径指定”，并单击“添加资源路径”分别在“路径”中输入.Trash 目录，例如 ***obs_bucket_name*/user/root/.Trash/***。
     ii. “bucket”选择“通过资源路径指定”，并单击“添加资源路径”在“路径”中输入 ***obs_bucket_name***。

     其中 **obs_bucket-name** 请使用实际的 OBS 桶名替换。
  e. （可选）请求条件，暂不添加。

图22-2 自定义策略

步骤 4 单击“确定”完成策略添加。

步骤 5 选择“委托”，并在步骤 1 中查询到的委托所在行的“操作”列单击“权限配置”。

步骤 6 查询并勾选步骤 3 中创建的策略。

步骤 7 单击“确定”完成委托权限配置。

步骤 8 重新执行 **hadoop fs -rm obs://<obs_path>**命令。

----结束