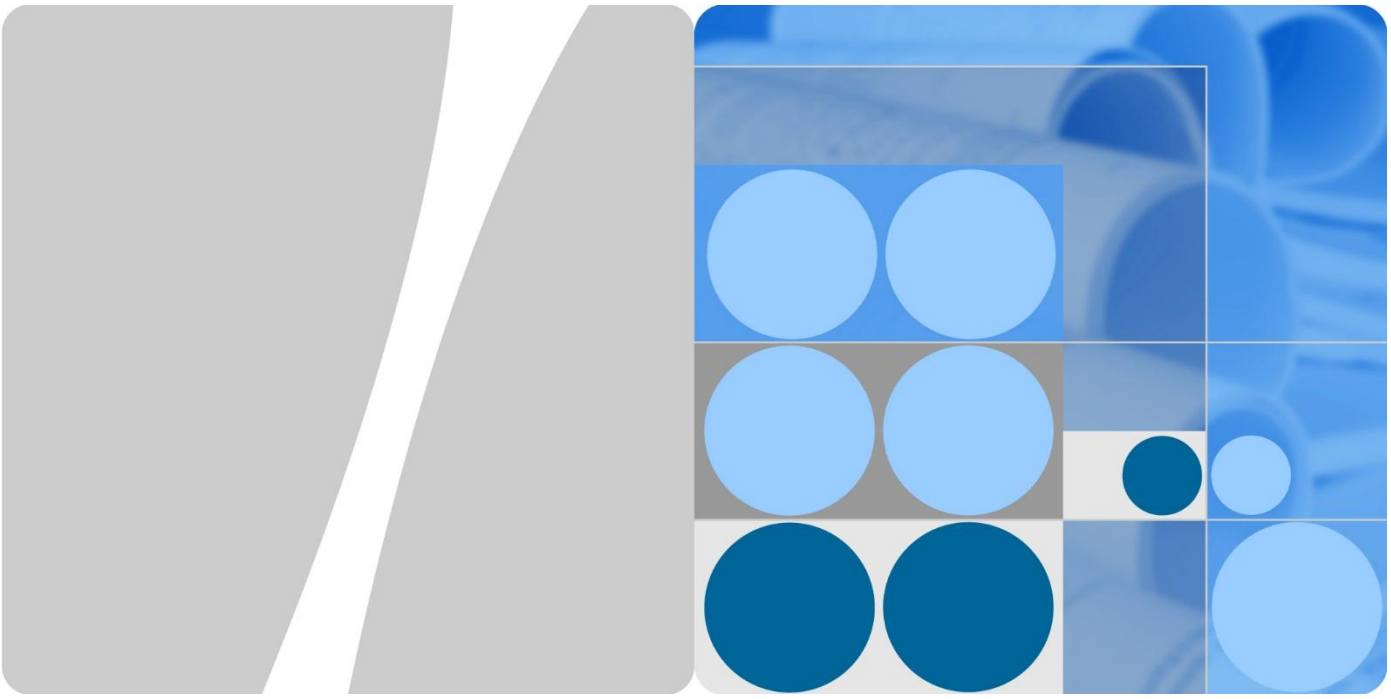


HBase 常见问题维护手册 V1.0

HBase 常见问题维护手册 V1.0

文档版本 01

发布日期 2016-03-31

华为技术有限公司

HUAWEI

商标声明


和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： http://www.huawei.com

客户服务邮箱： support@huawei.com

客户服务电话： 4008302118

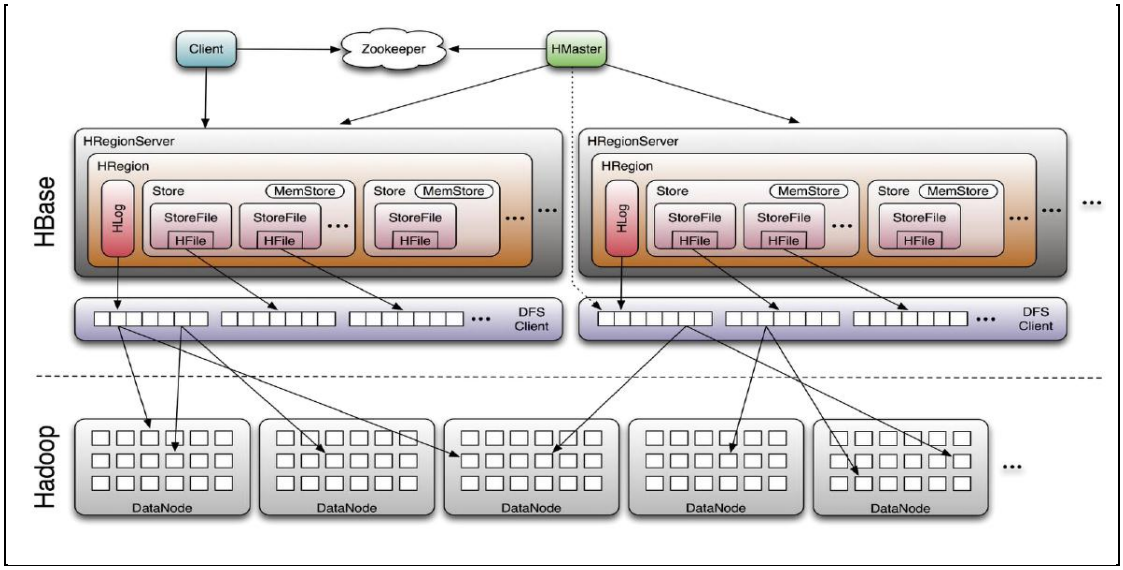
目 录

1、基本概念

【概述】

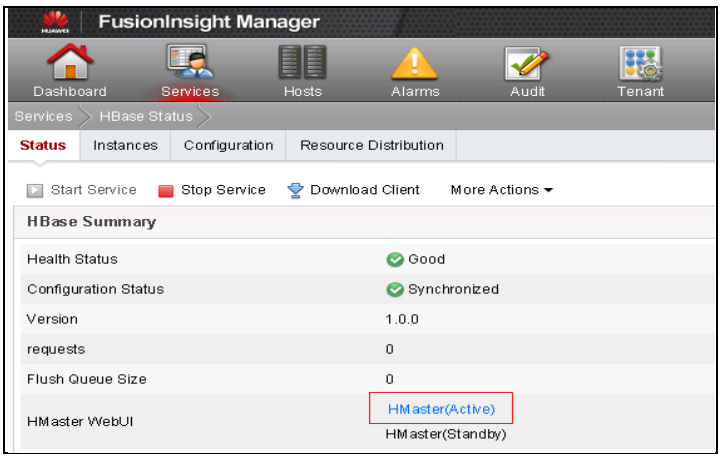
HBase 是一个高可靠性、高性能、面向列、可伸缩的分布式数据库，设计目标是用来解决关系型数据库在处理海量数据时的局限性。

架构图如下:

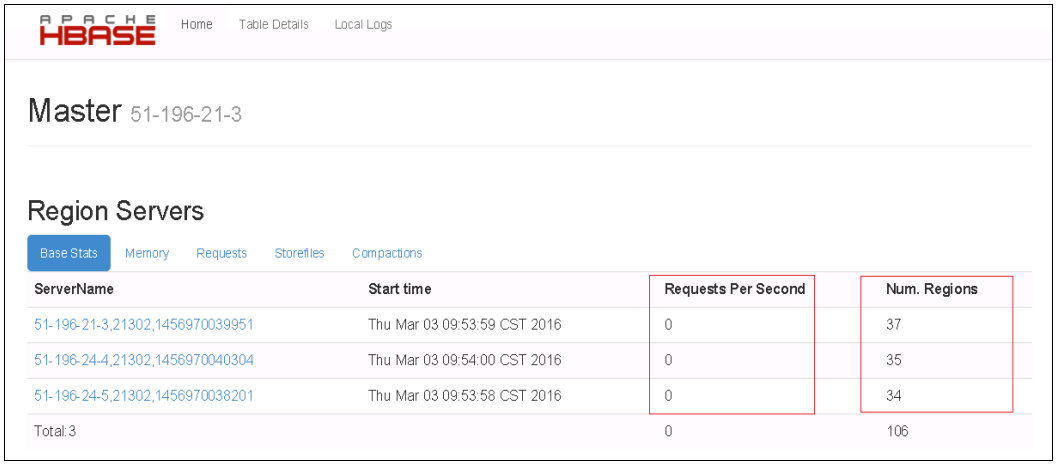


【HBase WebUI】

- 点击进入HBase WebUI



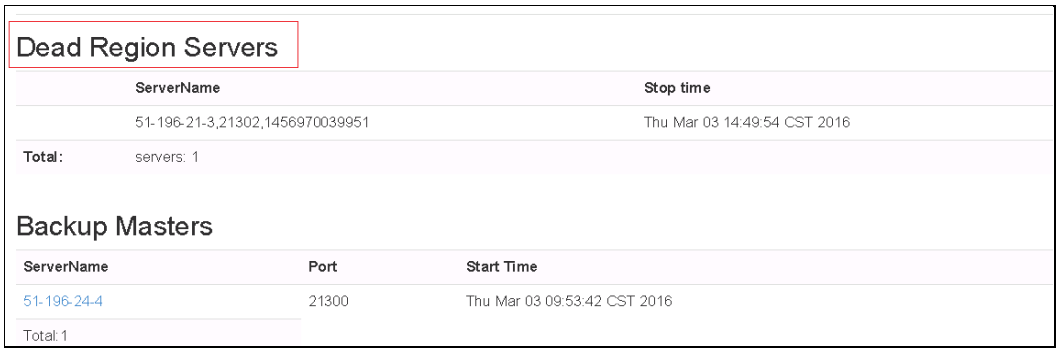
- 查看Region Server栏信息



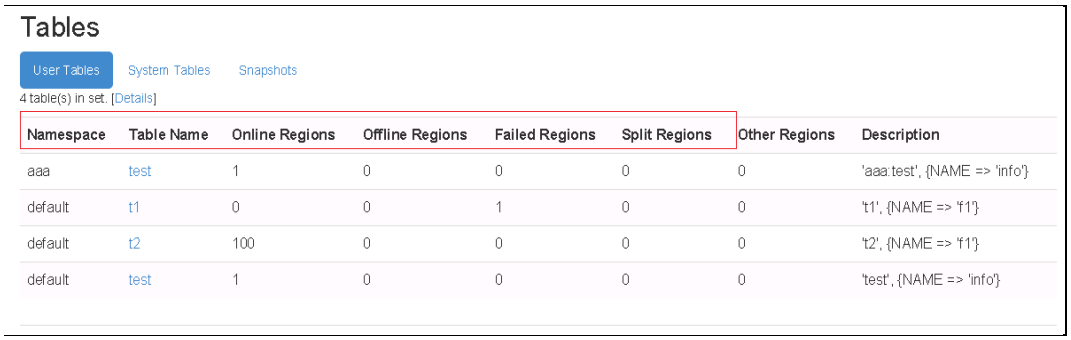
Requests Per Second，每秒读或写请求次数，可以用来监控 HBase 请求是否均匀。如果不均匀需排查是否为建表的 region 划分不合理造成。

Num. Regions，每个 Regionserver 节点上的 region 个数，观察每个节点的 region 个数是否均匀，如果不均匀需要确认 balance 问题。

- Dead Region Servers，查看Dead的节点并确认该节点是否是人为stop



- Tables栏信息



Tables栏分别有User Tables、System Tables、Snapshots。

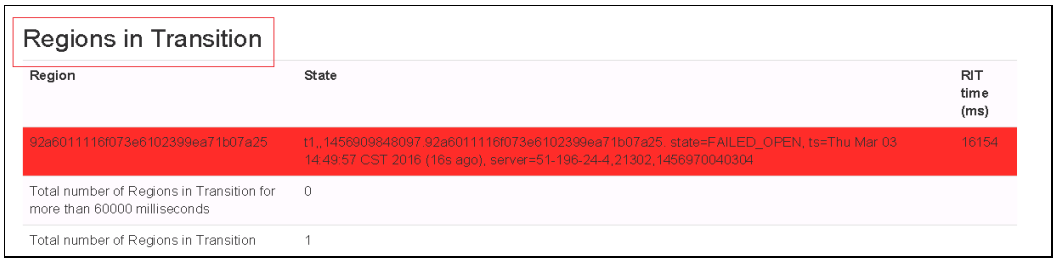
User Tables 记录用户创建表，可以查看到在线、下线、失败的 region 个数。Split Regions 记录进行自动 split 次数（重启 HMaster 后重新从 0 计算）。

System Tables 记录系统表，安全版本集群有三张系统表：hbase:acl、hbase:meta、hbase:namespace。如果 hbase:acl 表 region 未上线会导致 manager 页面对 HBase 授权失败，如果 hbase:namespace 表 region 未上线会导致创建表失败，hbase:meta 表未上线，会导致其

他 region 无法正常使用。

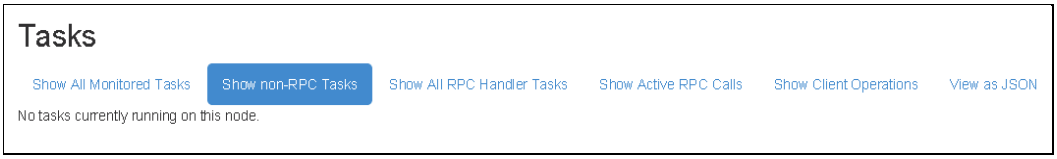
Snapshots 记录创建的快照信息。

- Regions in Transition栏信息



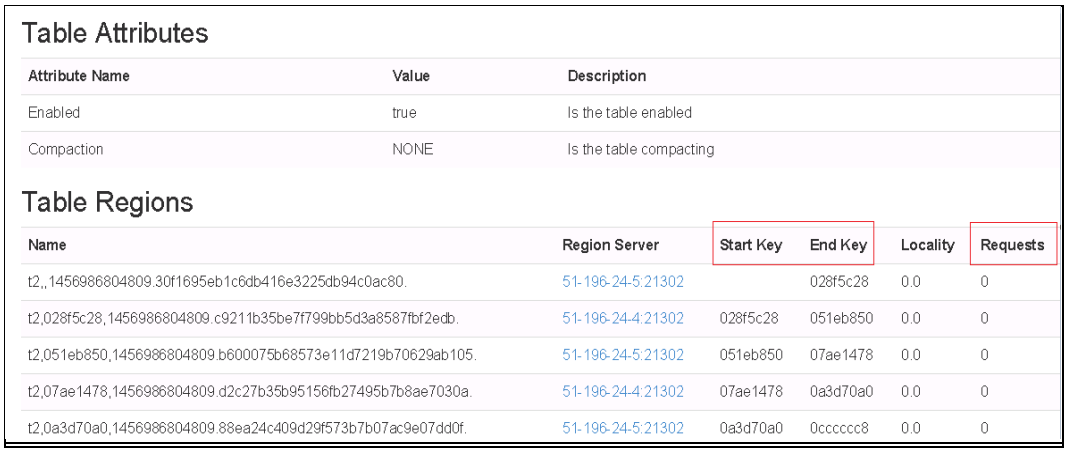
通过 Regions in Transition 栏可以查看到处于 RIT 中的事务，比如上图中 FAILED_OPEN region 事务。

- Tasks栏，一般用来与RIT栏一起确认，如果RIT为空且non-RPC为空则HBase服务启动正常。



- Table表详细信息

在 Tables 栏通过点击某一张表可以跳转至该页面，可以通过 Requests 查看是否请求均匀分别在多个 region 上。



每一个 region 分别有 Start Key 和 End Key,相邻两个 region 需要前一个 region 的 END Key 等于后一个 region 的 Start Key。

如果存在相邻两个 region 为[12, 23)、[34,45)，中间缺少[23, 34)，这种现象为 There is holes in Meta table。

如果存在相邻两个 region 为[12, 23) 、[12, 45)，两个 region 交叉，这种现象为 Region Overlap。

上述两种现象都为异常现象。

【如何确认 HBase 服务状态正常】

- ✧ 确认 Manager 页面显示启动成功并且每个实例都是 health
- ✧ 查看 HBase WebUI，查看是否一直存在 Dead Servers、Region in Transition 以及 Non-RPC Task 是否为空。如果一直存在，则 HBase 服务存在问题。
- ✧ 确认每个 Regionserver 节点上 region 个数是否均匀以及各节点 Requests 数是否均匀（可以不 check）
- ✧ 使用客户端检验 hbase 服务是否正常，检验方法如下:

  使用管理员用户认证（不记得密码可以使用 hbase.keytab 认证）

  source $Client_Home/bigdata_env

  kinit hbase

  hbase hbck

  上述“hbase hbck”指令会输出类似如下显示:

```
2016-03-03 16:29:12,750 INFO  [main-SendThread(51-196-21-3:24002)] zookeeper.ClientCnxn: Session establishment complete on server 51-19
 0xd004f0639273113, negotiated timeout = 90000
2016-03-03 16:29:12,766 INFO  [main] zookeeper.ZooKeeper: Session: 0xd004f0639273113 closed
2016-03-03 16:29:12,766 INFO  [main-EventThread] zookeeper.ClientCnxn: EventThread shut down
Summary:
Table hbase:meta is okay.
    Number of regions: 1
    Deployed on:  51-196-24-4,21302,1456970040304
Table hbase:acl is okay.
    Number of regions: 1
    Deployed on:  51-196-24-4,21302,1456970040304
Table t1 is okay.
    Number of regions: 1
    Deployed on:  51-196-24-5,21302,1456970038201
Table t2 is okay.
    Number of regions: 100
    Deployed on:  51-196-24-4,21302,1456970040304 51-196-24-5,21302,1456970038201
Table test is okay.
    Number of regions: 1
    Deployed on:  51-196-24-4,21302,1456970040304
Table aaa:test is okay.
    Number of regions: 1
    Deployed on:  51-196-24-5,21302,1456970038201
Table hbase:namespace is okay.
    Number of regions: 1
    Deployed on:  51-196-24-4,21302,1456970040304
0 inconsistencies detected.
Status: OK
2016-03-03 16:29:12,925 INFO  [main] client.ConnectionManager$HConnectionImplementation: Closing master protocol: MasterService
2016-03-03 16:29:12,925 INFO  [main] client.ConnectionManager$HConnectionImplementation: Closing zookeeper sessionid=0xe004f06cf4684ae
2016-03-03 16:29:12,937 INFO  [main] zookeeper.ZooKeeper: Session: 0xe004f06cf4684ae closed
2016-03-03 16:29:12,937 INFO  [main-EventThread] zookeeper.ClientCnxn: EventThread shut down
```

  最后会输出汇总信息，需要确认最后 Status 是否为 OK。

【日志概述】

日志路径：HBase 相关日志的默认存储路径为“/var/log/Bigdata/HBase”和“/var/log/Bigdata/audit/hbase”。

文件名	日志内容
checkServiceDetail.log	HBase 健康状态检查日志
hbase-omm-master-XXX.log	Hmaster 服务日志
hbase-omm-master-XXX.out	Hmaster 服务启动日志
master-om-gc.log	Hmaster GC 日志
hbase-omm-regionserver-XXX.log	RegionServer 服务日志
hbase-omm-regionserver-XXX.out	RegionServer 服务启动日志

regionserver-omm-gc.log	RegionServer GC 日志
hbase-audit-hmaster.log	Hmaster 审计日志
Hbase-audit-regionserver	RegionServer 审计日志

【客户端工具】

日常定位问题，常常需要使用客户端来连接 HBase，来进行一些测试验证，从而排除相关可能。

```
total 10124
drwxr-xr-x 5 root root     4096 Feb 29 10:18 HBase
drwxr-xr-x 6 root root     4096 Feb 29 10:18 HDFS
drwxr-xr-x 7 root root     4096 Feb 29 10:19 Hive
drwxr-xr-x 3 root root     4096 Feb 29 10:19 JDK
drwxr-xr-x 4 root root     4096 Feb 29 10:19 Kafka
drwxr-xr-x 4 root root     4096 Feb 29 10:19 KrbClient
drwxr-xr-x 4 root root     4096 Feb 29 10:19 Loader
drwxr-xr-x 4 root root     4096 Feb 29 10:19 SmallFS
drwxr-xr-x 6 root root     4096 Feb 29 10:19 Solr
drwxr-xr-x 5 root root     4096 Feb 29 10:19 Spark
drwxr-xr-x 4 root root     4096 Feb 29 10:20 Yarn
drwxr-xr-x 5 root root     4096 Feb 29 10:20 ZooKeeper
-rwxr-xr-x 1 root root      729 Feb 29 10:20 bigdata_env
-rwxr-xr-x 1 root root     4651 Feb 29 10:20 conf.py
-rwxr-xr-x 1 root root 10281273 Feb 29 10:20 jythonLib.jar
-rwxr-xr-x 1 root root     1799 Feb 29 10:20 switchuser.py
```

注意：下述 IP 地址信息，请根据实际情况进行修改。

- 建表语句

  建表语句可以参考下图所示，可以用默认参数建表或者设置某些属性（例如 VERSIONS、TTL），另外建表时候可以预分 Region（比如设置 SPLITS 等）

```
Create a table with namespace=ns1 and table qualifier=t1
  hbase> create 'ns1:t1', {NAME => 'f1', VERSIONS => 5}

Create a table with namespace=default and table qualifier=t1
  hbase> create 't1', {NAME => 'f1'}, {NAME => 'f2'}, {NAME => 'f3'}
  hbase> # The above in shorthand would be the following:
  hbase> create 't1', 'f1', 'f2', 'f3'
  hbase> create 't1', {NAME => 'f1', VERSIONS => 1, TTL => 2592000, BLOCKCACHE => true}
  hbase> create 't1', {NAME => 'f1', CONFIGURATION => {'hbase.hstore.blockingStoreFiles' => '10'}}

Table configuration options can be put at the end.
Examples:

  hbase> create 'ns1:t1', 'f1', SPLITS => ['10', '20', '30', '40']
  hbase> create 't1', 'f1', SPLITS => ['10', '20', '30', '40']
  hbase> create 't1', 'f1', SPLITS_FILE => 'splits.txt', OWNER => 'johndoe'
  hbase> create 't1', {NAME => 'f1', VERSIONS => 5}, METADATA => { 'mykey' => 'myvalue' }
  hbase> # Optionally pre-split the table into NUMREGIONS, using
  hbase> # SPLITALGO ("HexStringSplit", "UniformSplit" or classname)
  hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit'}
  hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit', CONFIGURATION => {'hbase.hregion.scan.loadColumnFamiliesOnDemand' => 'true'}}

You can also keep around a reference to the created table:

  hbase> t1 = create 't1', 'f1'

Which gives you a reference to the table named 't1', on which you can then
call methods.
```

- 删除表操作

  disable ‘表名’

  drop ‘表名’

- Put/Get操作

```
Here is some help for this command:
Put a cell 'value' at specified table/row/column and optionally
timestamp coordinates.  To put a cell value into table 'ns1:t1' or 't1'
at row 'r1' under column 'c1' marked with the time 'ts1', do:

  hbase> put 'ns1:t1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value', ts1
  hbase> put 't1', 'r1', 'c1', 'value', {ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1, {ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1, {VISIBILITY=>'PRIVATE|SECRET'}}

The same commands also can be run on a table reference. Suppose you had a reference
t to table 't1', the corresponding command would be:

  hbase> t.put 'r1', 'c1', 'value', ts1, {ATTRIBUTES=>{'mykey'=>'myvalue'}}
```

```
Here is some help for this command:
Get row or cell contents; pass table name, row, and optionally
a dictionary of column(s), timestamp, timerange and versions. Examples:

  hbase> get 'ns1:t1', 'r1'
  hbase> get 't1', 'r1'
  hbase> get 't1', 'r1', {TIMERANGE => [ts1, ts2]}
  hbase> get 't1', 'r1', {COLUMN => 'c1'}
  hbase> get 't1', 'r1', {COLUMN => ['c1', 'c2', 'c3']}
  hbase> get 't1', 'r1', {COLUMN => 'c1', TIMESTAMP => ts1}
  hbase> get 't1', 'r1', {COLUMN => 'c1', TIMERANGE => [ts1, ts2], VERSIONS => 4}
  hbase> get 't1', 'r1', {COLUMN => 'c1', TIMESTAMP => ts1, VERSIONS => 4}
  hbase> get 't1', 'r1', {FILTER => "ValueFilter(=, 'binary:abc')"}
  hbase> get 't1', 'r1', 'c1'
  hbase> get 't1', 'r1', 'c1', 'c2'
  hbase> get 't1', 'r1', ['c1', 'c2']
  hbsase> get 't1','r1', {COLUMN => 'c1', ATTRIBUTES => {'mykey'=>'myvalue'}}
  hbsase> get 't1','r1', {COLUMN => 'c1', AUTHORIZATIONS => ['PRIVATE','SECRET']}
```

- 查询表操作

```
Some examples:

  hbase> scan 'hbase:meta'
  hbase> scan 'hbase:meta', {COLUMNS => 'info:regioninfo'}
  hbase> scan 'ns1:t1', {COLUMNS => ['c1', 'c2'], LIMIT => 10, STARTROW => 'xyz'}
  hbase> scan 't1', {COLUMNS => ['c1', 'c2'], LIMIT => 10, STARTROW => 'xyz'}
  hbase> scan 't1', {COLUMNS => 'c1', TIMERANGE => [1303668804, 1303668904]}
  hbase> scan 't1', {REVERSED => true}
  hbase> scan 't1', {FILTER => "(PrefixFilter ('row2') AND
    (QualifierFilter (>=, 'binary:xyz'))) AND (TimestampsFilter ( 123, 456))"}
  hbase> scan 't1', {FILTER =>
    org.apache.hadoop.hbase.filter.ColumnPaginationFilter.new(1, 0)}
For setting the Operation Attributes
  hbase> scan 't1', { COLUMNS => ['c1', 'c2'], ATTRIBUTES => {'mykey' => 'myvalue'}}
  hbase> scan 't1', { COLUMNS => ['c1', 'c2'], AUTHORIZATIONS => ['PRIVATE','SECRET']}
For experts, there is an additional option -- CACHE_BLOCKS -- which
switches block caching for the scanner on (true) or off (false).  By
default it is enabled.  Examples:

  hbase> scan 't1', {COLUMNS => ['c1', 'c2'], CACHE_BLOCKS => false}

Also for experts, there is an advanced option -- RAW -- which instructs the
scanner to return all cells (including delete markers and uncollected deleted
cells). This option cannot be combined with requesting specific COLUMNS.
Disabled by default.  Example:

  hbase> scan 't1', {RAW => true, VERSIONS => 10}
```

- Assign/Unassign操作

```
Here is some help for this command:
Assign a region. Use with caution. If region already assigned,
this command will do a force reassign. For experts only.
Examples:

  hbase> assign 'REGIONNAME'
  hbase> assign 'ENCODED_REGIONNAME'


hbase(main):029:0> unassign

ERROR: wrong number of arguments (0 for 1)

Here is some help for this command:
Unassign a region. Unassign will close region in current location and then
reopen it again.  Pass 'true' to force the unassignment ('force' will clear
all in-memory state in master before the reassign. If results in
double assignment use hbck -fix to resolve. To be used by experts).
Use with caution.  For expert use only.  Examples:

  hbase> unassign 'REGIONNAME'
  hbase> unassign 'REGIONNAME', true
  hbase> unassign 'ENCODED_REGIONNAME'
  hbase> unassign 'ENCODED_REGIONNAME', true
```

- Split/Merge操作

```
hbase(main):020:0> split

ERROR: wrong number of arguments (0 for 1)

Here is some help for this command:
Split entire table or pass a region to split individual region.  With the
second parameter, you can specify an explicit split key for the region.
Examples:
    split 'tableName'
    split 'namespace:tableName'
    split 'regionName' # format: 'tableName,startKey,id'
    split 'tableName', 'splitKey'
    split 'regionName', 'splitKey'


hbase(main):021:0> merge_region

ERROR: wrong number of arguments (0 for 2)

Here is some help for this command:
Merge two regions. Passing 'true' as the optional third parameter will force
a merge ('force' merges regardless else merge will fail unless passed
adjacent regions. 'force' is for expert use only).

NOTE: You must pass the encoded region name, not the full region name so
this command is a little different from other region operations.  The encoded
region name is the hash suffix on region names: e.g. if the region name were
TestTable,0094429456,1289497600452.527db22f95c8a9e0116f0cc13c680396. then
the encoded region name portion is 527db22f95c8a9e0116f0cc13c680396

Examples:

  hbase> merge_region 'ENCODED_REGIONNAME', 'ENCODED_REGIONNAME'
  hbase> merge_region 'ENCODED_REGIONNAME', 'ENCODED_REGIONNAME', true
```

- Balancer/Balance_switch

  后面讲解

- Hbase shell重定向方式

  echo "scan 'hbase:meta'" | hbase shell> out

- 查看HFile

  hbase org.apache.hadoop.hbase.io.hfile.HFile -v -p -m –f HDSF 文件路径

- Bulkload指令（具体信息参考产品文档）

  第一步：

  hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.bulk.output=/tmp/bulkoutput -Dimporttsv.separator=',' -Dimporttsv.columns=HBASE_ROW_KEY,f:c1,f:c2 testBulkload /tmp/bulkdata

  第二步

  hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles </path/for/output> <tablename>

- Hbase hbck指令

  hbase hbck

  检查HBase集群是否异常

  hbase hbck –details

  检查并输出详细打印

  hbase hbck table1 table2

  只检查表table1和表table2

  hbase hbck –fixAssignments

  修复未上线、上线异常或者multiply assingned的region

  注意：**下列操作为高危操作**

  hbase hbck -fixAssignments –fixMeta

修复 region 同时修复 hbase:meta 表存在的异常信息

```
hbase hbck -repair TableFoo
```

修复表TableFoo

2、常见问题

注意：案例中 IP 地址信息，请根据实际情况进行修改。

【启动监控异常】

[HBase-10001] 端口被占用导致 Regionserver 启动失败。

【问题背景与现象】

Manager 页面监控发现 Regionserver 状态为 Concerning。

【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 查看到 pid，然后根据 pid 查看到相应的进程，发现 RegonServer 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 FI 产品端口范围重叠，因为安装时未进行 preinstall 操作。

【解决办法】

1. 手工 kill -9 DFSZkFailoverController 的 pid， 使得其重启后绑定其它端口，然后重启 concerning 的 regionserver。

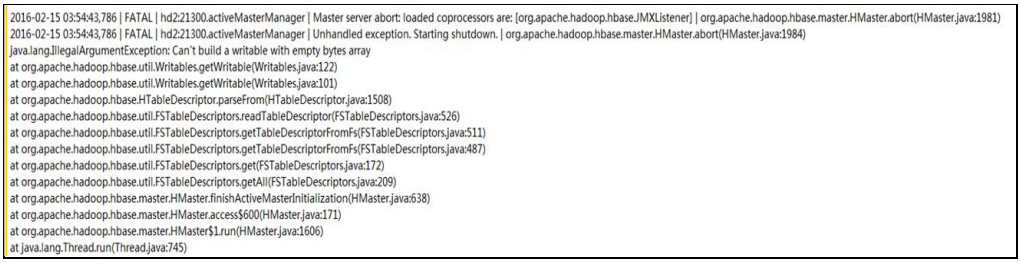
[HBase-10002] 表描述文件异常导致 HBase 启动失败。

【问题背景与现象】

使用 FusionInsight C50 版本，HBase 启动失败（底层数据存储使用 N9000）。

【原因分析】

1. 查看 HMaster 日志(/var/log/Bigdata/hbase/hm/hbase-omm-xxx.log),显示如下异常打印导致 HMaster 服务 abort。



2. 通过日志堆栈显示在读取表描述信息失败导致集群 abort（Can't build a writable with empty bytes array），
3. 通过使用客户端查看发现 HDFS 目录发现，存在某些表的.tableinfo.文件大小为 0。(下图为备份后的目录)

【解决办法】

1. 先将有问题的表描述文件 mv 到临时目录，然后启动 hbase 服务。
   hadoop fs –mv /hbase/data/default/表名/.tabledesc　/tmp
2. 成功启动 hbase 服务后，通过以下方式修复表描述文件：
   - 使用 hbase 管理员用户认证

     kinit hbase
   - 修复.tabledesc 文件

     hbase hbck –fixTableOrphans 表名
3. 重新检查该表是否正常

   hbase hbck 表名

   由于 fix 后的表描述为默认值，需要与之前建立表描述是否一致，查看修复后的表描述

   describe 表名

   如果与之前建立表描述不一致，则需要通过 alter 来进行修改。

[HBase-10003] 节点剩余内存不足导致 HBase 启动失败。

【问题背景与现象】

使用 FusionInsight C30 版本，HBase 的 regionserver 服务一直是 concerning 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】

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。

[HBase-10004] HDFS 性能差导致 HBase 服务不可用告警。

【问题背景与现象】

使用 FusionInsight C30SPC500 版本，HBase 组件断断续续上报服务不可用告警

【原因分析】

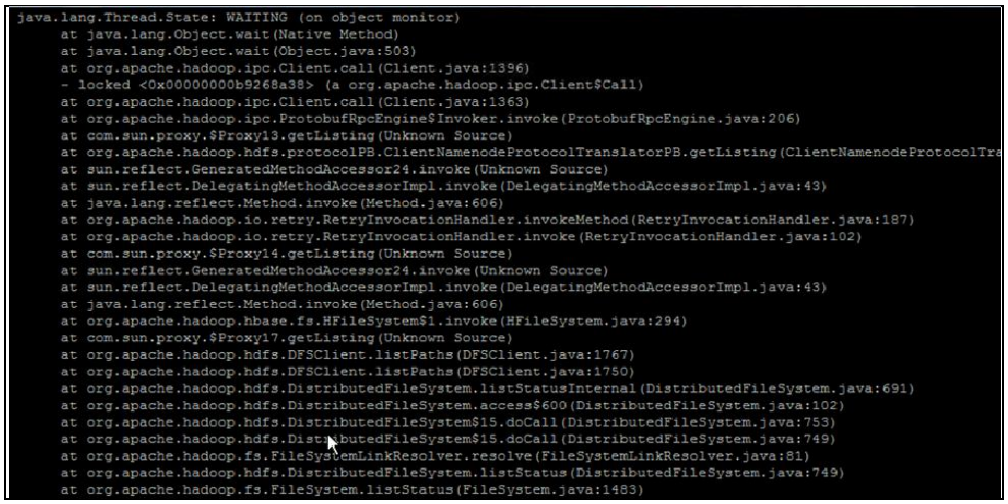
1. 该问题多半为 HDFS 性能较慢，导致健康检查超时，从而导致监控告警。可通过以下方式判断：
   - 首先查看 HMaster 日志（/var/log/Bigdata/hbase/hm/hbase-omm-xxx.log），确认 HMaster 日志中没有频繁打印"system pause"或"jvm"等 GC 相关信息。
   - 然后可以通过下列三种方式确认原因为 HDFS 性能慢造成告警产生：
     i. 使用客户端验证，通过 hbase shell 进入 hbase 命令行后，执行 list 验证需要运行多久。
     ii. 开启 HDFS 的 debug 日志，然后查看下层目录很多的路径（hadoop fs –ls /XXX/XXX），验证需要运行多久。
     iii. 打印 hmaster 进程 jstack

     su – omm

     Jps

     jstack pid

     iv. 如下图所示，Jstack 显示一直卡在 DFSClient.listPaths



【解决办法】

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)。

[HBase-10005] 参数不合理导致 HBase 启动失败。

【问题背景与现象】

使用模板安装 FusionInsight C50SPC200 版本集群，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是总和低于 0.8。

【解决办法】

1. 修改配置参数后，重启 hbase 服务成功。

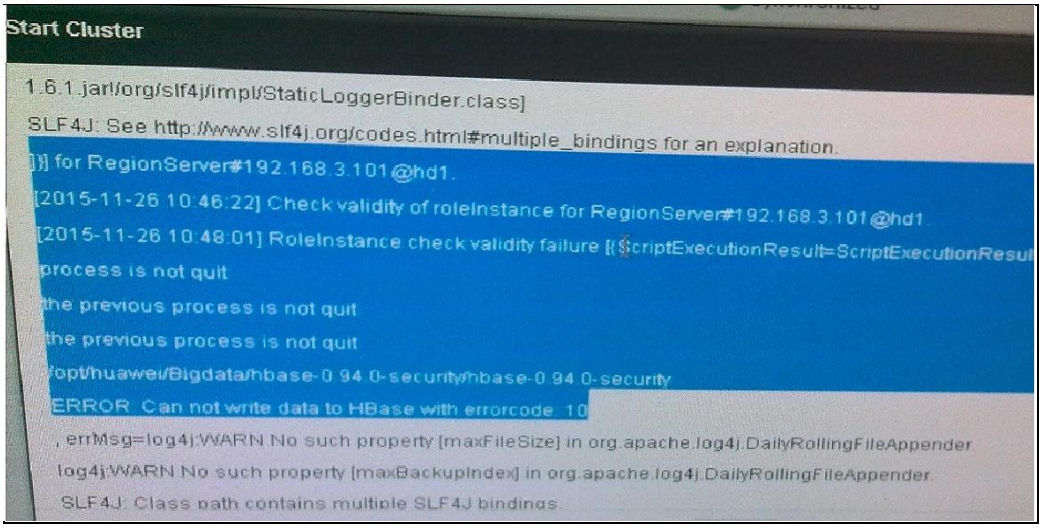
[HBase-10006] 残留进程导致 Regionsever 启动失败。

【问题背景与现象】

使用 FusionInsight C30SPC600 版本，HBase 服务启动失败，健康检查报错

【原因分析】

1. 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】

1. 登录 192.168.3.101 节点，后台通过 ps –ef | grep HRegionServer 发现后台确实存在一个残留的进程。
2. 确认进程可以 Kill 后，Kill 掉该进程（如果 kill 不掉需要考虑其他方法，比如重启该节点操作系统）
3. 重新启动 HBase 服务成功。

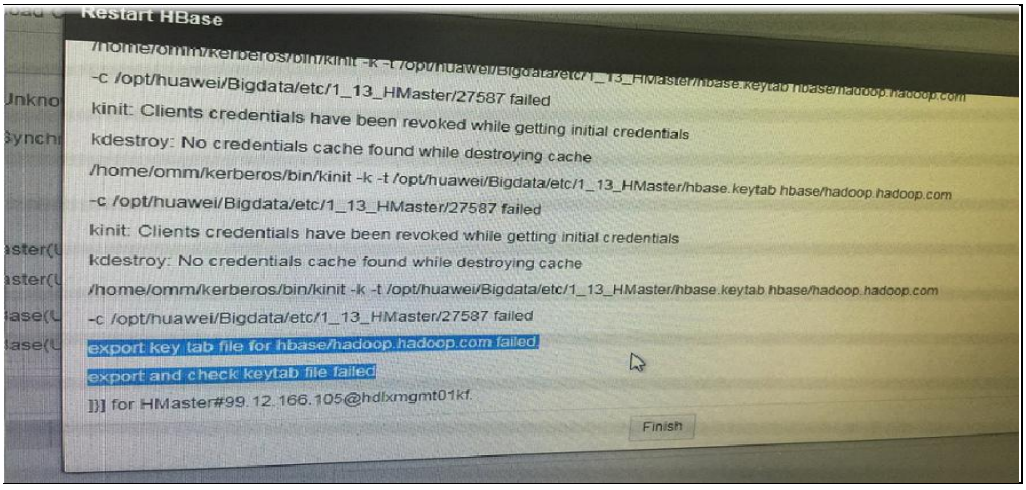
[HBase-10007] 管理员用户被锁导致 HBase 服务启动失败。

【问题背景与现象】

使用 FusionInsight C30SPC600 版本，HBase 组件启动一直不成功。

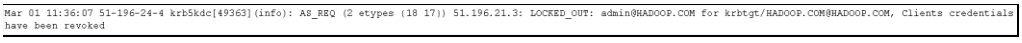
【原因分析】

1. 启动 HBase 时 Manager 页面提示相应信息为“Clients credentials have been revoked while gettinginital credentials”，HBase 管理员用户被锁，导致启动失败。



【解决办法】

1. 查看 kerberos 的 kdc 日志，发现在 HBase 启动失败时，还存在某个节点一直使用 hbase 用户进行访问。经过后续确认有开发应用使用错误的 HBase 的 keytab，替换为正确的 keytab 恢复正常。
2. 例如出现如下“Clients credentials have been revoked”，需要查看是否有 hbase 用户被锁（下图是以 admin 用户举例）。



[HBase-10008] 磁盘空间不足导致 HBase 启动失败。

【问题背景与现象】

使用 FusionInsight C02SPC300 版本，HBase 启动失败。

【原因分析】

1. 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】

1. 通过后台使用 df –h 指令查看，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回复。
2. 后续需要扩容硬盘或者节点来解决数据目录空间不足问题。

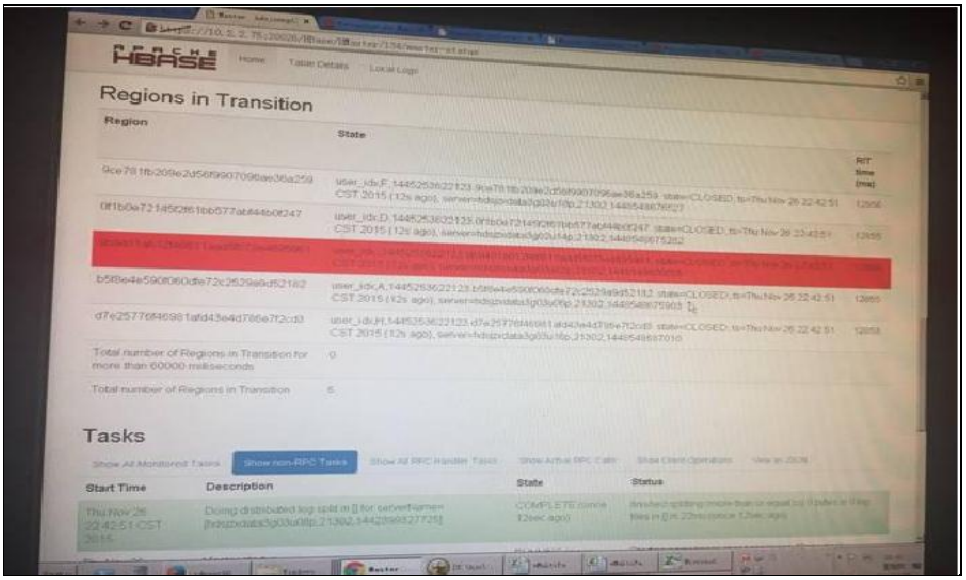
[HBase-10009] HBase 的 initial 超时导致启动失败。

【问题背景与现象】

使用 FusionInsight C50 版本，HBase 启动失败提示初始化超时。

【原因分析】

1. 因为之前残留大量 RIT 事务，而参数 hbase.master.initializationmonitor.haltontimeout 默认 300 秒，该时间内还未来得及完成初始化，导致 Hmater 进程主动退出。



【解决办法】

1. 将参数 hbase.master.initializationmonitor.haltontimeout 增大（比如临时改成 30 分钟）
2. 重启 HBase 服务成功。

[HBase-10010] HBase version 文件损坏导致启动失败。

【问题背景与现象】

使用 FusionInsight C30SPC100 版本，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 文件不能正常查看，该文件损坏。

【解决办法】

1. 从本地同版本集群中获取 hbase.version 文件上传进行替换，
2. 重新启动 HBase 服务成功。

[HBase-10011] Session control 导致 RegionServer 一直 concerning。

【问题背景与现象】

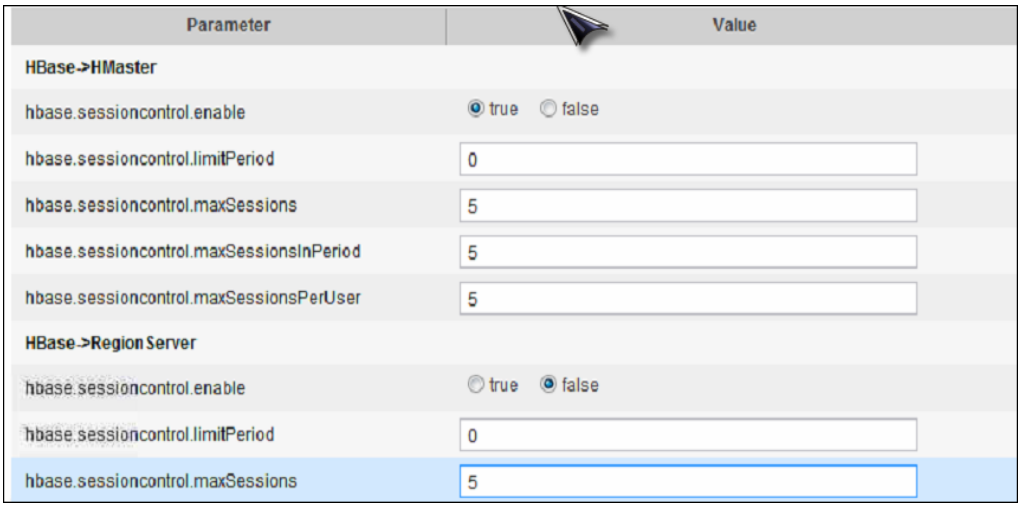
使用 FusionInsight C02SPC503 版本，有一个 Regionserver 一直处于 concerning 状态。

【原因分析】

1. 通过 Regionserver 日志显示如下异常，Overflow maximum number of peruser limit：

```
2016-03-04 13:29:31,957 WARN  [regionserver26003] Unable to connect to master. Retrying. Error was:
org.apache.hadoop.hbase.regionserver.HRegionServer.getMaster(HRegionServer.java:1897)
org.apache.hadoop.hbase.DoNotRetryIOException: Overflow maximum number of peruser limit: 5
```

2. 查看 Manager 配置的 Session Control



【解决办法】

1. 与现场沟通关闭 Session Control（上面 hbase.sessioncontrol.enable 设置为 false），
2. 重启 HBase 服务，regionserver 恢复正常。

【Region 不在线与其他应用异常】

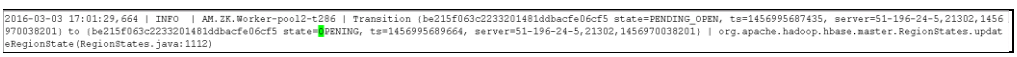
[HBase-20001] 磁盘空间满导致 region 上线失败。

【问题背景与现象】

通过 HBase WebUI 发现存在部分 region 未成功上线。

【原因分析】

1. 查看 Hmater 日志信息，可以根据 region 名称找到下图这种日志打印，确认该 region assign 到哪一个 regionserver。



2. 对应 regionserver 节点后，查看 regionserver 日志。日志提示没有可用的 datanode。
3. 通过 df –i 指令查看，发现数据目录磁盘 inode 已满。

【解决办法】

1. 与现场沟通删除不需要的数据后，重启 HBase 服务恢复。

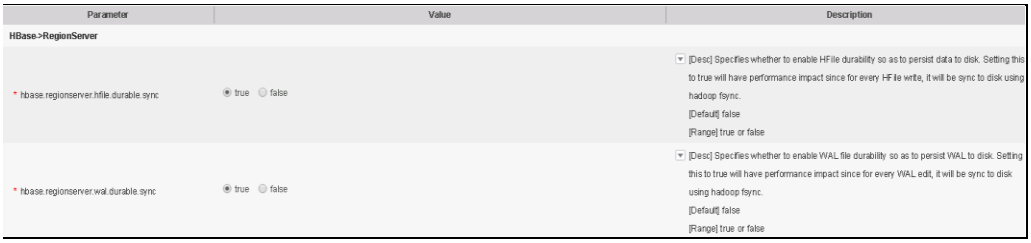
[HBase-20002] Sync 功能导致 HBase 入库性能下降。

【问题背景与现象】

升级到 FusionInsight C50 版本，HBase 入库性能比 C02、C30 版本差很多。

【原因分析】

1. C50 版本将如下两个参数默认值设置为 true，关闭 sync 功能可以提供 HBase 入库性能。



【解决办法】

2. 现场经过沟通确认，关闭 sync 功能，性能极大提升。

【扩展说明】

如果是高价值零容忍的数据，是不建议关闭 sync 功能的。
如果场景是对性能需求最高，也能一定程度容忍数据丢失，才建议关闭 sync 功能。

举例说明丢数据的可能性为：

- 一条数据没有在写完成 HLog 和内存之后，内存中的数据固化成 HFile 之前，如果 HLog 所在的 HDFS 文件，该文件（或部分数据块）3 副本数的所有节点宕机了，刚好这些数据块还在 OS 的缓存中，就有可能丢数据。
- 一条数据固化成了 HFile，该 HFile 文件（或部分数据块）在 HDFS 中 3 副本数所有节点宕机了，并且这些数据块还在 OS 的缓存中，就有可能丢数据。

[HBase-20003] 使用不同过滤查询方式性能不同。

【问题背景与现象】

当前使用如下两种不同方式，需要查询时间不同。
第一种方式：

```
new RowFilter(CompareFilter.CompareOp.EQUAL, new BinaryPrefixComparator(Bytes.toBytes(rowkeyStartStr)));
```

第二种方式：

```
scan.setRowPrefixFilter(Bytes.toBytes(“xxxx”))
```

【原因分析】

1. 第一种方法采用 BinaryPrefixComparator，如果没有同时设置 StartRow 和 StopRow，又因为当前 filter 条件查询不到数据，因此需要遍历 region 的所有数据。
2. 第二种方法 setRowPrefixFilter 是通过设置 StartRow 和 StopRow 来实现。
   例如 scan.setRowPrefixFilter（Bytes.toBytes(“123”)），那么 StartRow 为 123，StopRow 会自动计算为 124，那么查询是只需要查看 123~124 这段范围，因此需要的时间比第一种短。

【解决办法】

1. 当前使用第一种方法，则同时设置 StartRow 和 StopRow。

[HBase-20004] HBase 客户端写线程较多时，查询业务缓慢。

【问题背景与现象】

客户端写线程较多时，查询业务缓慢，如果将写线程停止后，查询业务正常。

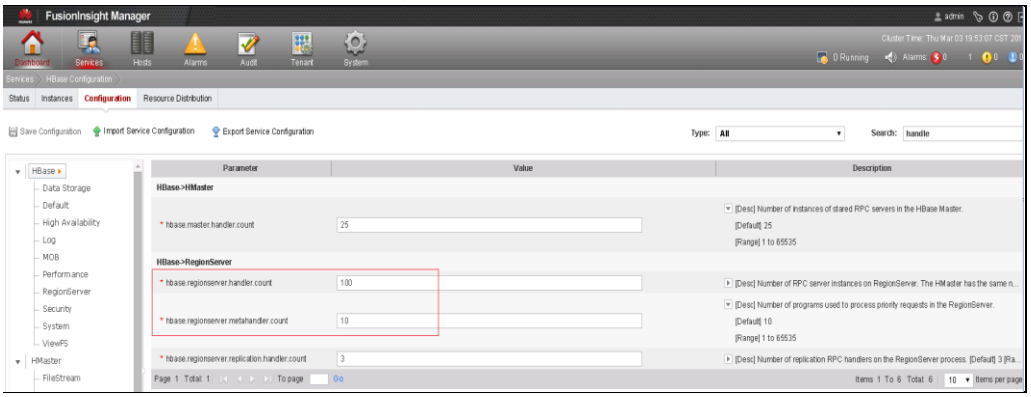
【原因分析】

1. 通过打印 regionserver 的 jstack 信息。
   发现写任务全是使用 PriorityRPCServer.handler 而不是 B.defaultRpcServer.handler。

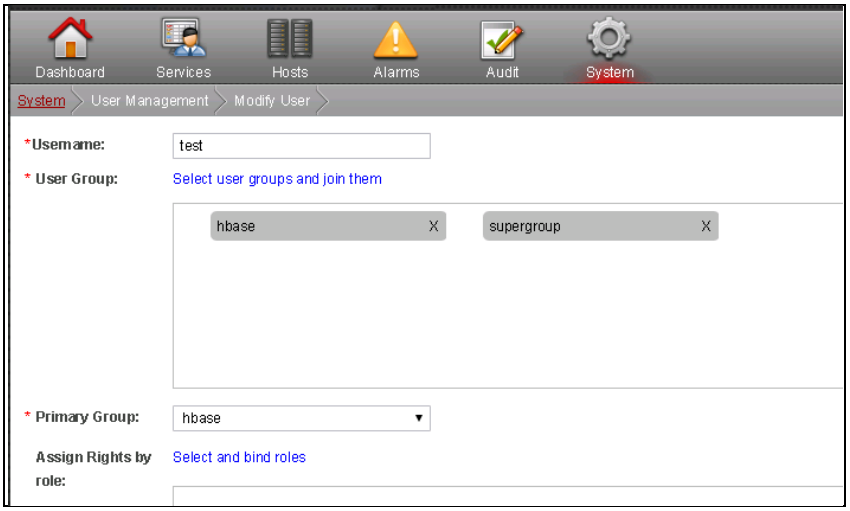
```
"PriorityRpcServer.handler=1,queue=1,port=21302" #254 daemon prio=5 os_prio=0 tid=0x00007f3ed0da9800 nid=0x26b23 in Object.wait() [0x00007f3ebb937000]
   java.lang.Thread.State: TIMED_WAITING (on object monitor)
 at java.lang.Object.wait(Native Method)
 at org.apache.hadoop.hbase.regionserver.wal.SyncFuture.get(SyncFuture.java:167)
 - locked <0x000000065212f348> (a org.apache.hadoop.hbase.regionserver.wal.SyncFuture)
 at org.apache.hadoop.hbase.regionserver.wal.FSHLog.blockOnSync(FSHLog.java:1468)
 at org.apache.hadoop.hbase.regionserver.wal.FSHLog.publishSyncThenBlockOnCompletion(FSHLog.java:1462)
 at org.apache.hadoop.hbase.regionserver.wal.FSHLog.sync(FSHLog.java:1596)
 at org.apache.hadoop.hbase.regionserver.HRegion.syncOrDefer(HRegion.java:8146)
 at org.apache.hadoop.hbase.regionserver.HRegion.doMiniBatchMutation(HRegion.java:3357)
 at org.apache.hadoop.hbase.regionserver.HRegion.batchMutate(HRegion.java:2939)
 at org.apache.hadoop.hbase.regionserver.HRegion.batchMutate(HRegion.java:2874)
 at org.apache.hadoop.hbase.regionserver.HRegion.batchMutate(HRegion.java:2878)
```

```
 at
org.apache.hadoop.hbase.regionserver.RSRpcServices.doBatchOp(RSRpcServices.java:702)
 at
org.apache.hadoop.hbase.regionserver.RSRpcServices.doNonAtomicRegionMutation(RSRpcServices.java:662)
 at
org.apache.hadoop.hbase.regionserver.RSRpcServices.multi(RSRpcServices.java:2091)
```

2. PriorityRPCServer 为系统使用，DefaultRpcServer 为用户使用，当前开发应用使用 supergroup 的 keytab 导致一直使用 PriorityRPCServer，而 PriorityRPCServer 的线程数只有 10 个，在写线程较多的情况下，没有足够 handler 来处理读取操作。



3. 通过用户管理界面确认是否添加 supergroup 用户组



【解决办法】

1. 新建一个 keytab 来进行使用，而不是 supergroup 用户。
2. 建议不要使用 supergroup 用户访问 HBase。

【二次开发问题】

[HBase-30001] Kerberos 用户被锁导致开发应用运行失败。

【问题背景与现象】

开发应用访问 FusionInsight C30SPC600 版本 HBase 组件有时成功有时失败,相应异常信息如下:

```
2015-07-14 12:59:08,674 WARN [[ACTIVE] ExecuteThread: '4' for queue: 'weblogic.kernel.Default (self-tuning)'] zookeeper.ZKUtil: hconnection-0x248ecefb-0xd4e88f9d2ac02e3, quorum=T101PC03VM13:24002,T101PC03VM12:24002,T101PC03VM14:24002, baseZNode=/hbase Unable to set watcher on znode (/hbase/hbaseid)
org.apache.zookeeper.KeeperException$SessionExpiredException: KeeperErrorCode = Session expired for /hbase/hbaseid
  at org.apache.zookeeper.KeeperException.create(KeeperException.java:127)
  at org.apache.zookeeper.KeeperException.create(KeeperException.java:51)
  at org.apache.zookeeper.ZooKeeper.exists(ZooKeeper.java:1045)
```

【原因分析】

1. 查看 Kerberos 日志（/var/log/Bigdata/Kerberos/krb5kdc.log），发现日志中显示当前开发应用使用的 keytab 用户一直被锁，用户被锁信息打印类似下图(have been revoked)

```
Mar 01 11:36:07 51-196-24-4 krb5kdc[49363](info): AS_REQ (2 etypes {18 17}) 51.196.21.3: LOCKED_OUT: admin@HADOOP.COM for krbtgt/HADOOP.COM@HADOOP.COM, Clients credentials have been revoked
```

【解决办法】

1. 根据 Kerberos 日志中显示的 ip，找到访问来源，排查到有另一个应用程序也使用同一个用户访问，经过排查发现该应用 keytab 有问题从而导致用户被锁。

[HBase-30002] 多次重复登录导致 24 小时后应用异常。

【问题背景与现象】

开发应用访问 C30SPC600 版本 HBase 组件运行一段时间后出现异常，异常信息如下：

```
15-6-26 17:00:00:146 CST] 000000b7 SecureClient E org.apache.hadoop.hbase.ipc.SecureClient$SecureConnection$1 run SASL authentication failed. The most likely cause is missing or invalid credentials. Consider 'kinit'.
javax.security.sasl.SaslException: Failure to initialize security context [Caused by org.ietf.jgss.GSSException，主代码：8，次代码：0
主字符串：凭证已过期
次字符串：Kerberos 凭证已过期]
    at com.ibm.security.sasl.gsskerb.GssKrb5Client.<init>(GssKrb5Client.java:131)
```

```
        at com.ibm.security.sasl.gsskerb.FactoryImpl.createSaslClient(FactoryImpl.java:53)
        at javax.security.sasl.Sasl.createSaslClient(Sasl.java:362)
        at
org.apache.hadoop.hbase.security.HBaseSaslRpcClient.<init>(HBaseSaslRpcClient.java:97)
        at
org.apache.hadoop.hbase.ipc.SecureClient$SecureConnection.setupSaslConnection(Secu
reClient.java:169)
```

【原因分析】

1. 通过客户端日志发现其中多次打印“Login successful for”，因此该问题为重复登录问题。

```
[15-6-25 1:46:53:077 CST] 00000060 UserGroupInfo I
org.apache.hadoop.security.UserGroupInformation loginUserFromKeytab Login
successful for user ...
[15-6-25 2:46:53:092 CST] 00000060 UserGroupInfo I
org.apache.hadoop.security.UserGroupInformation loginUserFromKeytab Login
successful for user ...
[15-6-25 3:46:53:107 CST] 00000060 UserGroupInfo I
org.apache.hadoop.security.UserGroupInformation loginUserFromKeytab Login
successful for user ...
```

【解决办法】

1. 修改客户端代码，同一个进程中只进行一次 kerberos 认证。

[HBase-30003] 错误配置文件导致开发应用运行失败。

【问题背景与现象】

开发应用访问 FusionInsight C30SPC600 版本 HBase 组件，异常信息如下

```
Tue Dec 08 16:14:13 CST 2015,
  org.apache.hadoop.hbase.client.RpcRetryingCaller@2aea5f5d,       java.io.IOException:
Couldn't setup connection for cmridmpsh@HADOOP.COM to
hbase/shyp-bigdata-dmp-dn12@HADOOP.COM
Tue Dec 08 16:14:44 CST 2015,
org.apache.hadoop.hbase.client.RpcRetryingCaller@2aea5f5d,       java.io.IOException:
Couldn't setup connection for cmridmpsh@HADOOP.COM to
  hbase/shyp-bigdata-dmp-dn12@HADOOP.COM
Tue Dec 08 16:15:15 CST 2015,
  org.apache.hadoop.hbase.client.RpcRetryingCaller@2aea5f5d, java.io.IOException:
  Couldn't setup connection for cmridmpsh@HADOOP.COM to
  hbase/shyp-bigdata-dmp-dn12@HADOOP.COM
```

【原因分析】

1. 查看日志信息显示 principal 为 hbase/主机名这种，而这种在 C30TR5 版本才存在，在 C30TR6 版本及后续版本改为 hbase/hadoop.hadoop.com。因此应用程序使用旧版本的 hbase-site.xml 配置文件。

【解决办法】

1. 应用程序中使用当前集群的配置文件后，问题解决。
2. 此类访问不成功问题，建议排查 jar 包是否正常、配置文件是否正确、keytab 及用户名是否正确、/etc/hosts 是否配置映射关系等。

[HBase-30004] 获取 HBase 数据失败，提示 RowTooBigException。

【问题背景与现象】

开发应用访问 FusionInsight C30SPC600 版本 HBase 组件，提示 RowTooBigException 异常信息。

【原因分析】

1. FusionInsight C50SPC200 版本 hbase.table.max.rowsize 设置为 1073741824。
2. 存在某条数据过长导致出现该异常，通过排查发现客户程序写入一条超过 1G 的 row。

【解决办法】

1. 修改 hbase.table.max.rowsize。
2. 删除超过 1G 的 row 的数据（可选）。

【咨询问题】

[HBase-40001] Bulkload 导入数据，region 未自动 split。

【解决办法】

1. Region 自动 split 只有通过 flushRegion 或者 Compaction 进行触发，如果只通过 bulkload 写入数据，则不会调用 flushRegion，因此只能通过 Compaction 触发自动 split。
2. Compaction 操作分为 minor compaction 和 major compaction，两个都是有由 CompactionChecker 的定时任务来触发，其中 minor compaction 每 10*1000 秒进行一次检查（storefile 数量大于等于 3 个），majorcompation 频率由 hbase.hregion.majorcompaction 控制，默认为 1 天。
3. 合并时需要检查 storefile 的大小是否大于 hbase.hstore.compaction.max.size，如果小于则这类 storefile 会被过滤掉而不进行 compaction。因为某些场景不能触发 compaction 或者合并后剩余 storefile 数量大于等于 hbase.hstore.blockingStoreFiles(默

认 7)，所以后续不进行自动 split。

[HBase-40002] 数据老化问题（TTL）。

【解决办法】

1. 关于 HBase 表设置 TTL 后，什么时候数据老化问题，默认方式建表后，TTL 为 FOREVER 即不会老化。

```
hbase(main):003:0> describe 't3'
Table t3 is ENABLED
t3
COLUMN FAMILIES DESCRIPTION
{NAME => 'info', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION =>
 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.0200 seconds
```

2. 当数据到底 TTL 设置时间后，老化的数据会在 minor compaction 中被删除。

> 39. Time To Live (TTL)
>
> ColumnFamilies can set a TTL length in seconds, and HBase will automatically delete rows once the expiration time is reached. This applies to *all* versions of a row - even the current one. The TTL time encoded in the HBase for the row is specified in UTC.
>
> Store files which contains only expired rows are deleted on minor compaction. Setting `hbase.store.delete.expired.storefile` to `false` disables this feature. Setting minimum number of versions to other than 0 also disables this.

[HBase-40003] HBase 写业务缓慢或者超时。

【解决办法】

1. 查看 HBase WebUI 发现 Requests 主要集中在少数的 Regionserver 上，该为热点现象，主要是 Rowkey 设计不合理导致。
   - 关于 Rowkey 设计可参考 HBase 官网 https://hbase.apache.org/book.html#rowkey.design

     1. **Salting** in this sense has nothing to do with cryptography, but refers to adding random data to the start of a row key.
     2. Instead of a random assignment, you could use a one-way **hash** that would cause a given row to always be "salted" with the same prefix.
     3. A third common trick for preventing hotspotting is to **reverse** a fixed-width or numeric row key so that the part that changes the most often (the least significant digit) is first.
   - 关于预分多少个 Region 可参考 HBase 官网 https://hbase.apache.org/book.html#arch.regions.size

     In general, HBase is designed to run with a small (20-200) number of relatively large (5-20Gb) regions per server.（只是建议值）

[HBase-40004] 如何查看 put 后的中文数据。

【解决办法】

```
get ‘表名’,’ rowkey’,{COLUNM=>[ ‘列族:列限定符:c(org.apache.hadoop.hbase.util.Bytes).toString’ ]}
```

```
hbase(main):017:0*
hbase(main):018:0* get 'zw_test','1',{COLUMN=>['info:name:c(org.apache.hadoop.hbase.util.Bytes).toString']}
COLUMN                                  CELL
 info:name                              timestamp=1455622748887, value=陈欣
1 row(s) in 0.0300 seconds
```

[HBase-40005] 如何对表进行重命名。

【解决办法】

1. 使用 Snapshot 功能对表重命名

138. Table Rename

In versions 0.90.x of hbase and earlier, we had a simple script that would rename the hdfs table directory and then do an edit of the hbase:meta table replacing all mentions of the old table name with the new. The script was called ./bin/rename_table.rb. The script was deprecated and removed mostly because it was unmaintained and the operation performed by the script was brutal.

As of hbase 0.94.x, you can use the snapshot facility renaming a table. Here is how you would do it using the hbase shell:

```
hbase shell> disable 'tableName'
hbase shell> snapshot 'tableName', 'tableSnapshot'
hbase shell> clone_snapshot 'tableSnapshot', 'newTableName'
hbase shell> delete_snapshot 'tableSnapshot'
hbase shell> drop 'tableName'
```

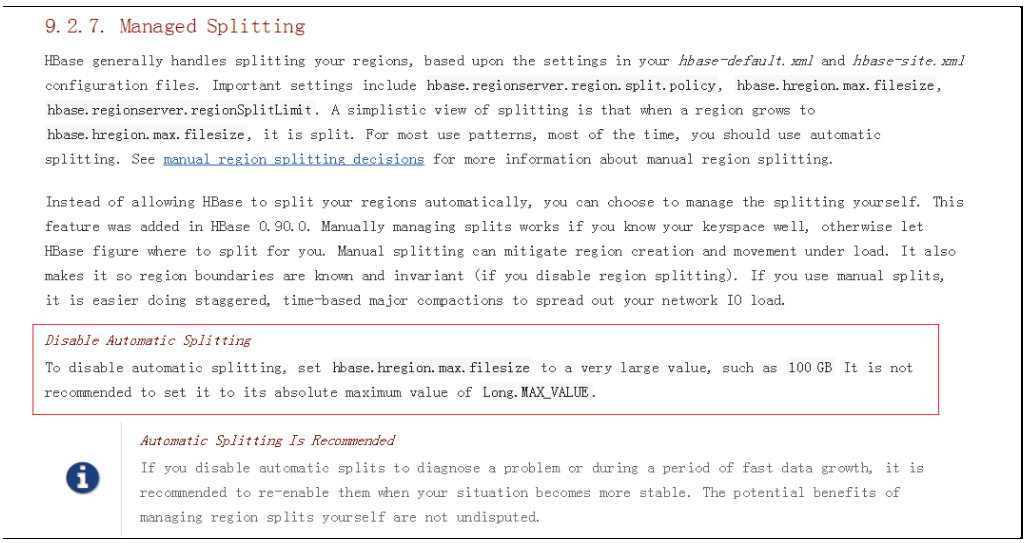
or in code it would be as follows:

```
void rename(Admin admin, String oldTableName, TableName newTableName) {
  String snapshotName = randomName();
  admin.disableTable(oldTableName);
  admin.snapshot(snapshotName, oldTableName);
  admin.cloneSnapshot(snapshotName, newTableName);
  admin.deleteSnapshot(snapshotName);
  admin.deleteTable(oldTableName);
}
```

[HBase-40006] 修改 hregion.max.filesize 后仍然自动 split region。

【解决办法】

1. 通过在建表时设置 hregion.max.filesize 为 Long.MAX_VALUE，但是依然能出现自动 split。
2. C50 默认采用 IncreasingToUpperBoundRegionSplitPolicy 策略，如果 hregion.max.filesize 设置为 Long.MAX_VALUE，上限值可能会计算为负数。
3. 根据官网介绍，只需要设置为一个较大值就可以，当前设置为 20T 后没有发生自动 split。



9.2.7. Managed Splitting

HBase generally handles splitting your regions, based upon the settings in your *hbase-default.xml* and *hbase-site.xml* configuration files. Important settings include `hbase.regionserver.region.split.policy`, `hbase.hregion.max.filesize`, `hbase.regionserver.regionSplitLimit`. A simplistic view of splitting is that when a region grows to `hbase.hregion.max.filesize`, it is split. For most use patterns, most of the time, you should use automatic splitting. See manual region splitting decisions for more information about manual region splitting.

Instead of allowing HBase to split your regions automatically, you can choose to manage the splitting yourself. This feature was added in HBase 0.90.0. Manually managing splits works if you know your keyspace well, otherwise let HBase figure where to split for you. Manual splitting can mitigate region creation and movement under load. It also makes it so region boundaries are known and invariant (if you disable region splitting). If you use manual splits, it is easier doing staggered, time-based major compactions to spread out your network IO load.

Disable Automatic Splitting

To disable automatic splitting, set `hbase.hregion.max.filesize` to a very large value, such as `100 GB` It is not recommended to set it to its absolute maximum value of `Long.MAX_VALUE`.

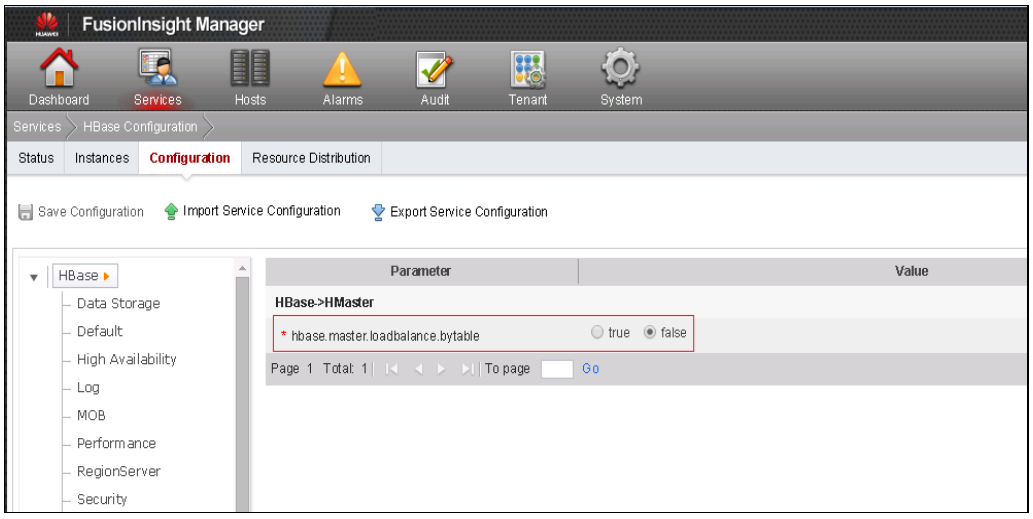
Automatic Splitting Is Recommended

If you disable automatic splits to diagnose a problem or during a period of fast data growth, it is recommended to re-enable them when your situation becomes more stable. The potential benefits of managing region splits yourself are not undisputed.

[HBase-40007] Balance 问题。

【解决办法】

1. 当前版本默认对 Regionserver 上 region 数来进行 balance，而不是对表进行 balance（有时会预计一张表有 100 个 region，但是大多数 region 都集中在一个 regionserver 节点上）。



2. 当 HBase 服务中存在 Region in Transition 或者 Dead Region servers 都不会进行触发 Balance。
3. 触发 Balance 后并还需要判断当前 region 是否分别不均匀。
   使用 hbase.regions.slop 默认值 0.2，判断是否存在某个 rs 的 region 数大于 average * 1.2 或者某个 rs 的 region 数小于 average * 0.8。
4. 可以通过指令来确认当前定时 balance 任务是否关闭。

```
source $Client_HOME/bigdata_env
kinit hbase
```

balance_switch true

5. 该指令将 balance 定时任务切换为 true 并返回上一次的状态，如下图所示，切换前是开启了 balance 服务的。

```
hbase(main):013:0* balance_switch true
true
0 row(s) in 0.0180 seconds
```

6. 如果 Balance 开启并且无 RIT 和 Dead Region server 情况，仍然没有触发 Balance，可以现场沟通是否可以开启 Hmaster 日志的 debug 级别。如果可以，则开启后查看日志中是否有打印“Not running balancer because”。

【Phoenix】

[HBase-50001] phoenix 查询超时问题。

【问题背景与现象】

使用 phoenix 访问 FusionInsight C30SPC100 版本 HBase 组件，运行 select 语句 10 分钟后报错，日志截图如下，还伴随有 rpc.timeout 的提示

```
        ... 15 more
Caused by: org.apache.hadoop.hbase.ipc.RemoteWithExtrasException(org.apache.hadoop.hbase.exceptions.OutOfOrderScannerNextException): (
.OutOfOrderScannerNextException: Expected nextCallSeq: 1 But the nextCallSeq got from client: 0; request=scanner_id: 286 number_of_row
call_seq: 0
        at org.apache.hadoop.hbase.regionserver.HRegionServer.scan(HRegionServer.java:3125)
```

【原因分析】

这个错误是因为 scan 的时候 socket 超时，因为 phoenix 只是 hbase 的客户端实现，所以基本上 hbase 不擅长的工作 phoenix 也是不擅长，例如聚合类操作。

1. 针对这个问题，可以执行这两步，将 scan 的这两个超时相关配置设长：

■ 修改客户端配置hbase-site.xml：

配置参数	默认值
phoenix.query.timeoutMs	60000
hbase.rpc.timeout	60000
hbase.client.scanner.timeout.period	60000

■ 修改服务端配置（在 om 上配置，需重启 HBase）：

配置参数	默认值
hbase.rpc.timeout	60000
hbase.client.scanner.timeout.period	60000

2. 这个只是暂时解决目前这个数据量下的超时，如果以后数据量增大，这个超时有可能也还要继续调整，由于这个超时跟数据量数据结构等都有关联，暂时还无法给出

一个定量的建议值。

【解决办法】

1. 按照上述修改客户端和服务端参数后并重启 HBase 服务解决。
2. 修改后还存在问题，则需要清空 system.stats 表。

[HBase-50002] Unable to find cached index metadata 问题。

【问题背景与现象】

使用 phoenix 访问 FusionInsight C50SPC200 版本 HBase 组件，运行一段时间后出现异常。

【原因分析】

1. 客户端日志显示如下异常信息：

```
2016/01/21 00:39:48 - GJ.6 - java.sql.SQLException: ERROR 2008 (INT10): Unable to find cached index metadata.  ERROR 2008 (INT10): ERROR 2008 (INT10): Unable to find cached index metadata. key=-691536508347998718 region=GJ,391,1452407521666.f422a5a462b3ef7a8c4f73849a75cc45. Index update failed
```

【解决办法】

2. 根据 https://issues.apache.org/jira/browse/PHOENIX-1718 单描述，建议服务端修改参数

```
<property>
<name>phoenix.coprocessor.maxServerCacheTimeToLiveMs</name>
<value>1800000</value>
</property>
<property>
<name>phoenix.coprocessor.maxMetaDataCacheTimeToLiveMs</name>
<value>1800000</value>
</property>
```