APACHE HBASE × MI

HBASECON ASIA2019

THE COMMUNITY EVENT FOR APACHE HBASE™

HBASECON ASIA2019

HBase Practice In China Mobile

Yechao Chen
China Mobile (Suzhou) Software Technology Co., Ltd.
China Mobile Suzhou Research Center

About China Mobile

1. China Mobile is the world's biggest telecom companies in the world
2. 932 million customers
3. 727 million 4G customers
4. 172 million wireline broadband customers
5. Over 100 PB data generated per day

About CMSoft

China Mobile Suzhou Software Technology Co., Ltd /China Mobile Suzhou Research Center.
Specialized subsidiary of China Mobile.
CMSoft focus on cloud computing ,big data and IT support related software services.

01 HBase on China Mobile

BC-Hadoop Architecture

High throughput
Real Time
Processing
Storage
Data Collection
AI
Analyze & Query
Streaming
Manager & Security
Tensor Flow
BigDL
Tez/ LLAP
Spark SQL
Presto
Phoenix
Spark Streaming
Flink
Storm /JStorm
BC-HControl
Spark
Hive
HBase
Pulsar
Kafka
YARN
ZooKeeper
BookKeeper
HDFS
Kerberos
HTunnel
Flume
Ranger

HBase Scales

1. Nodes：6000+ nodes
2. Clusters：100+ clusters，largest cluster with 600+ nodes
3. Data：tens of PBs，max table with 1.6 PB & 20000+Regions
4. Peak QPS: 30 millions rows/second with about 300 nodes

HBase Timeline

~2014

0.9.x

SQL on HBase
Secondary Index

2015

1.0.0

new api

lots of fix
POWER Server support

2016.3

1.1.3

stable and bug fix
quota

Id/key authentication
Sampling auto-split

2017.9

1.2.6

stable and bug fix
Java 8

Storage Policy support
SSD Accelerated
Phoenix support
Bulkload improve

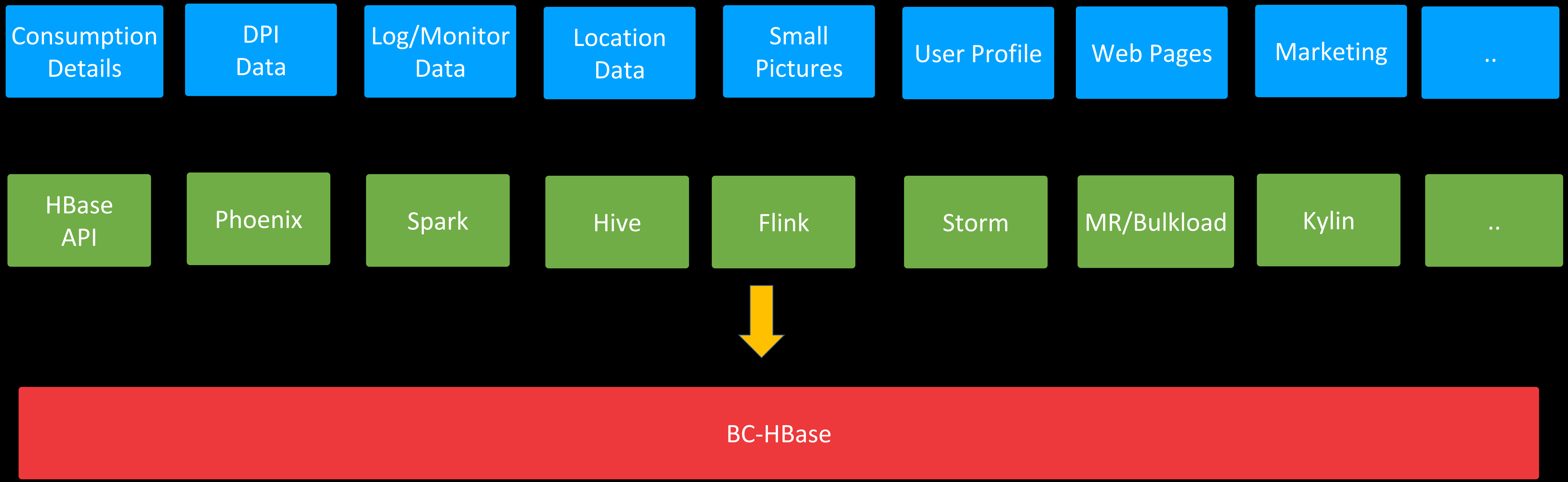
2019

2.1.2

RSG/MOB/hbck2
New Assignment Manager
more Read/Write path offheap

more permission support
ARM support
PFGA/ASIC Accelerated
Compression improve

Application Scenarios



02 Write Path Improve

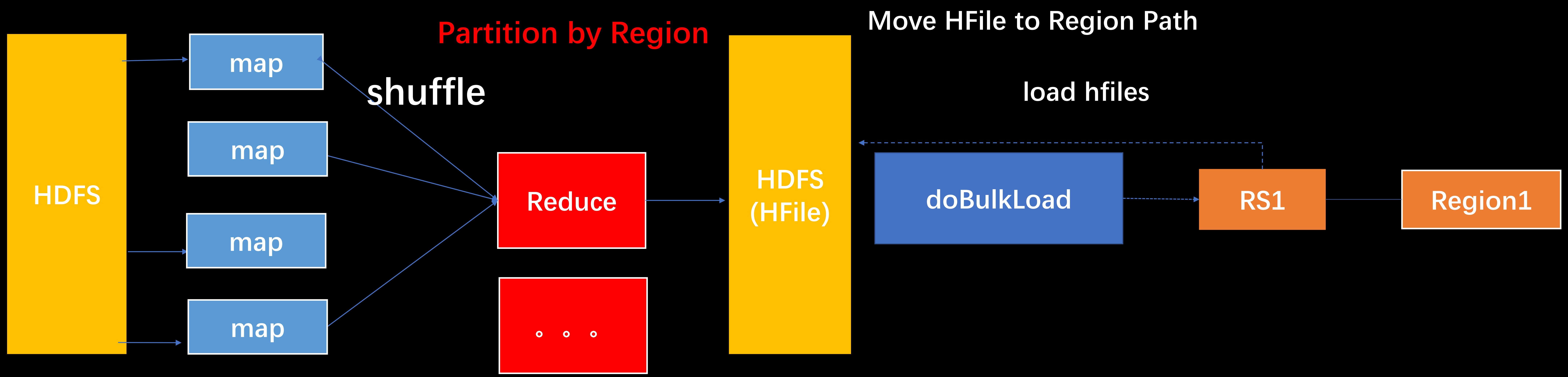
Write path improve

1. Bulkload with pre-split table by sampling
2. Bulkload HFile Data locality
3. Compression data write path improve
4. SSD Accelerated

Bulkload data skew problem

bulkload steps:

- Map: split HDFS Data to many splits,one map handle one split
- Reduce : task partition by table region startKeys, one reduce task create one region’s data
- doBulkload :move the HFile to region dir and update the HRegion HFile list



Bulkload data problem

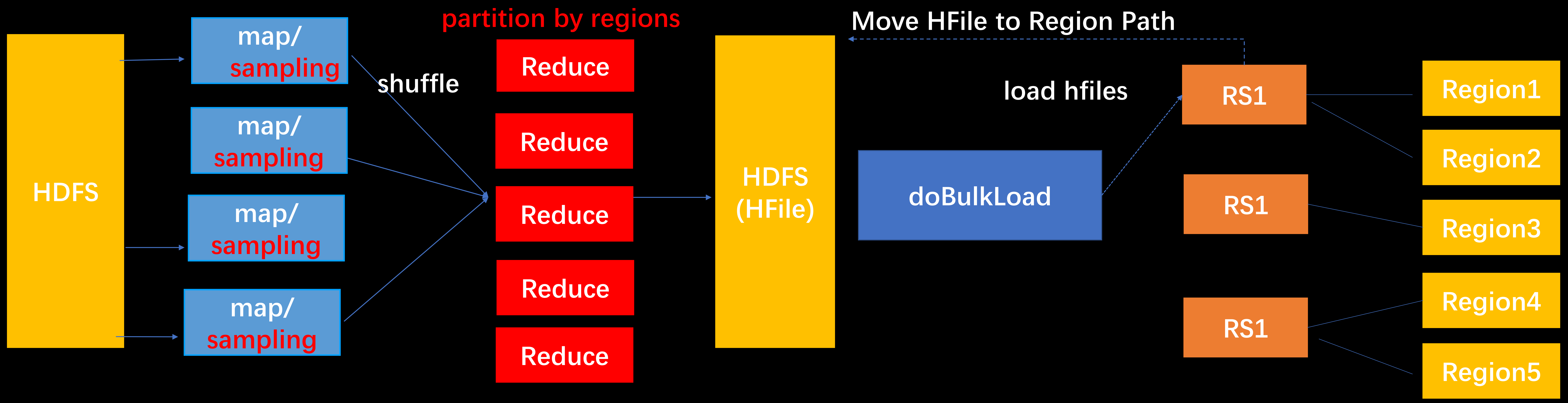
- table only one region by default
- data skew case poor performance
- hard to pre-split table or need to change the key to rowkey
- application need to change the code to match the new key(rowkey)
- why we need change the application sql or code ?

ID IN OTHERS	ROWKEY IN HBase	BEFORE	AFTER
139****1234	4321****931	USERID=139****1234	USERID=4321****931

Bulkload data skew improve

bulkload steps:

- Map: split HDFS Data to many splits,one map handle one split
- data smapling and pre-split by smapling data ,no need change the application code or sql any more!
- Reduce : task partition by table region startKeys,   one reduce task create one region's HFile data
- doBulkload :move the HFile to region dir and update the HRegion HFile list



Bulkload HFile data locality

bulkload HFile Data locality Problem:

- Bulkload HFile data locality is by reduce task ,not by the RegionServer of the Region
- Bulkload HFile data locality too low,more network traffic with read/compaction

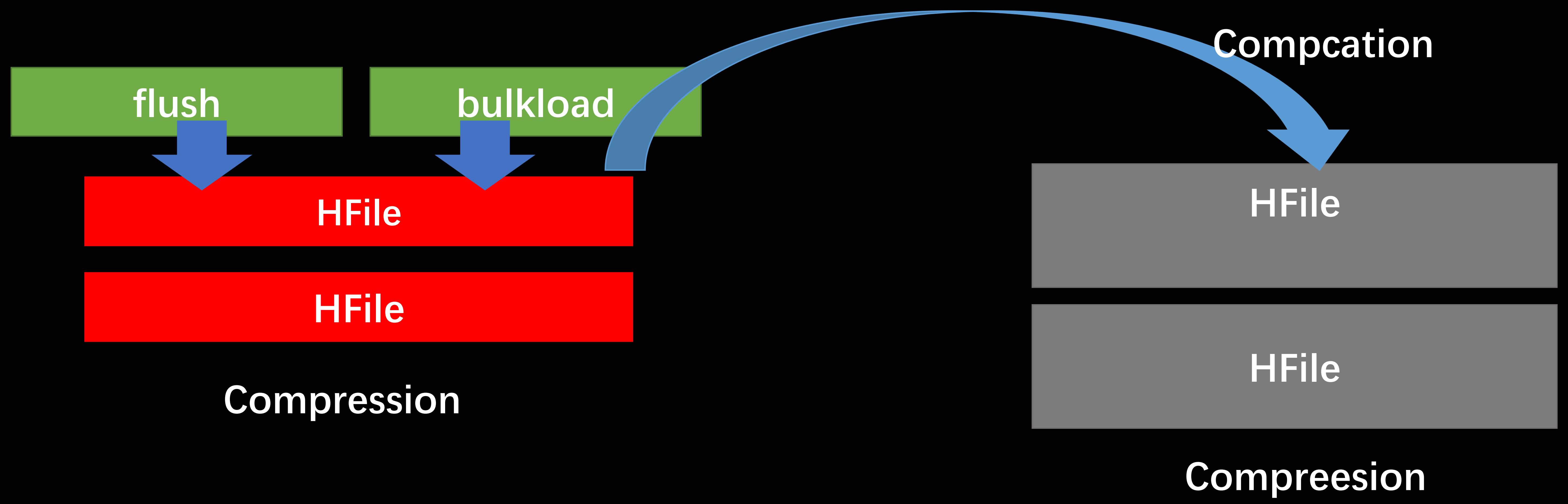
Improve:

- Bulkload Reduce task create one replica of the HFile on the RegionServer of the Region

Compress table problem

- Money is good : compression is a good choice to save the storage cost
- But flush MemStore to HFile with compression can reduce performance and cost more cpu
- bulkload to compression table slow than none-compression table also
- Life is short : the small HFiles by flush or bulkload will be merged by compaction

LSM Tree really always need compression???



Compress table improve

HOW

- Flush and bulkload HFile use none compression type : write path with no compression cost at all
- Compreesion just happened in compaction

Improve:

- Access first small hfiles before compaction is fast same as none compression
- Compcation will merge small hfiles to big hfiles with compression finally
- First Compaction the small HFiles more faster with no decompression cost
- also works for DATA_BLOCK_ENCODING

SSD Accelerated with write path

backgroup:

- Node :12*6TB HDD + 1*1.6TB PCIe SSD
- HDD:SSD =45:1
- How to use ssd more effective ?

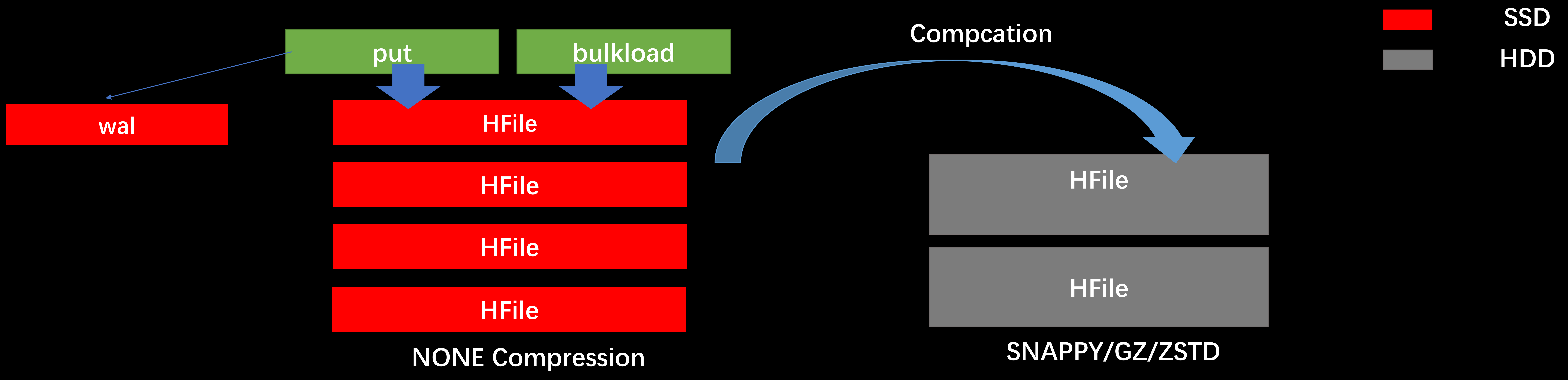
Improve:

- Backport HSM to our 1.2.6 Version
- WAL on ALL_SSD First : hbase.wal.storage.policy=ALL_SSD
- HFile first created by bulkload or flush on ALL_SSD but table storage type is HOT(HDD)
- ALL user write path happened in SSD,more faster than HDD
- Small HFiles before compaction in SSD is good for reading and compaction

Storage Policy

- hot or import table data in ALL_SSD or ONE_SSD
- SSD table region should keep more HFiles than HDD before minor compaction
  - too much compaction can reduce the ssd life
  - SSD random read is far faster than hdd
- ONE_SSD bug found: HDFS-14512

SSD and Compression



Jira and Config

Jira

●HBASE-12596 (bulkload needs to follow locality)
●HBASE-21810(bulkload support set hfile compression on client)
●HBASE-6572(Tiered HFile storage)
●HBASE-20105(Allow flushes to target SSD storage)
●HDFS-14512(ONE_SSD policy will be violated while write data with DistributedFileSystem.create(….favoredNodes)

Config

```
hbase.wal.storage.policy=ALL_SSD
create ‘test’, {NAME => ‘f’, CONFIGURATION => {‘hbase.hstore.flush.storagepolicy' => 'ALL_SSD’},
COMPRESSION => 'NONE', METADATA => {'COMPRESSION_COMPACT' => 'GZ'}}
Bulkload : -Dhbase.hstore.block.storage.policy=ALL_SSD -
Dhbase.mapreduce.hfileoutputformat.compression=none
```

03 Others

Replication

Backgroup

- replication will happened in two different data center
- user use bulkload not put api
- bandwidth limit
- RegionServer failed restart when add peer config cluster key error

Improve:

- Support HFile Bulkload Replication
- support set bulkload HFile compression ,reduce the HFile transmission bandwidth from two data center
- bug fix

Replication Related Jira

- HBASE-13153(Bulk Loaded HFile Replication)
- HBASE-21810(bulkload support set hfile compression on client)
- HBASE-15769(Perform validation on cluster key for add_peer)

Multi Tenant

- Isolation: Slider vs RegionServerGroup

	RegionServerGroup	Slider
Isolation	Physical isolation	Base on YARN (vcores and memory)
Use Case	online service/import service	offline or less import service
Manager	less clusters easy to manager	many clusters hard to manager

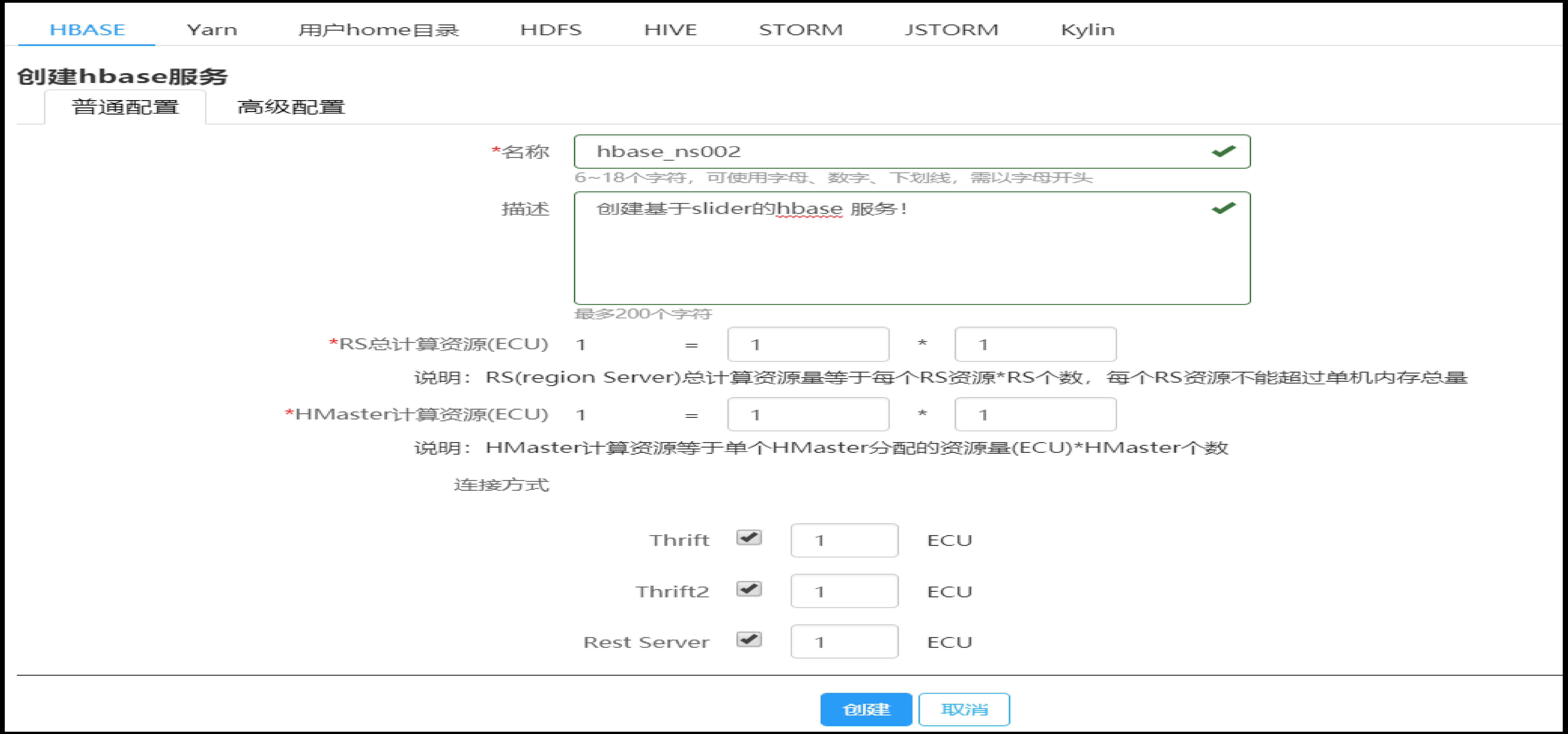
TIPS:

create a group to hanlder meta table

HBASECON ASIA2019

Multi Tenant

HBase on Slider



创建hbase服务

*名称 hbase_ns002

6~18个字符，可使用字母、数字、下划线，需以字母开头

描述 创建基于slider的hbase 服务！

最多200个字符

*RS总计算资源(ECU) 1 = 1 * 1

说明：RS(region Server)总计算资源量等于每个RS资源*RS个数，每个RS资源不能超过单机内存总量

*HMaster计算资源(ECU) 1 = 1 * 1

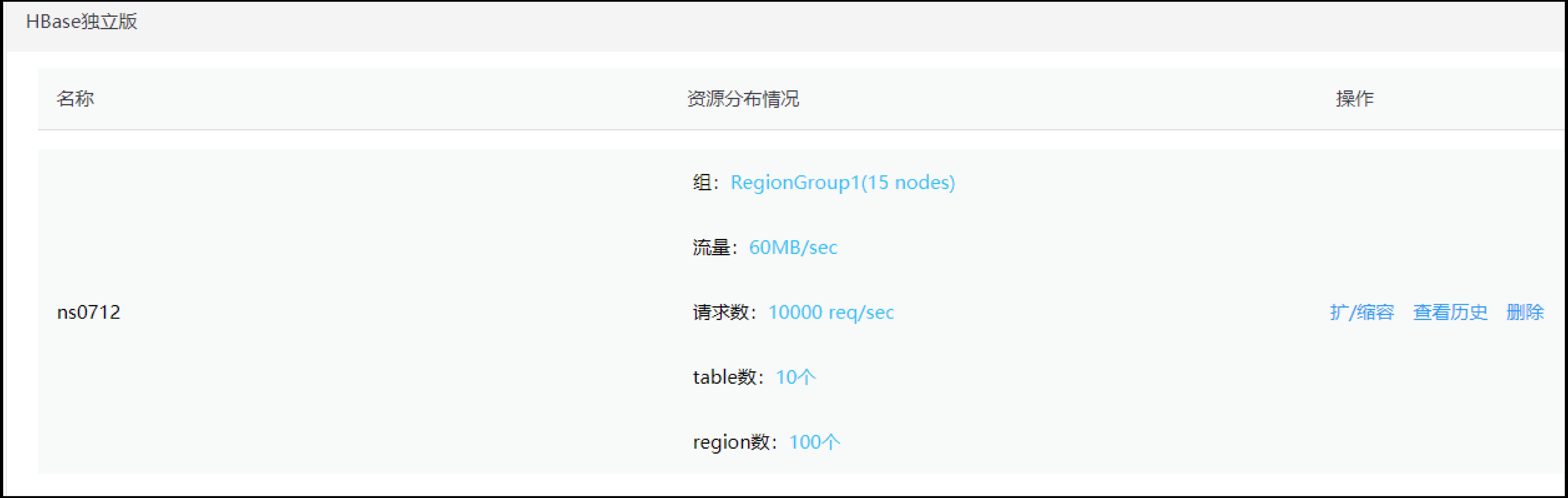
说明：HMaster计算资源等于单个HMaster分配的资源量(ECU)*HMaster个数

连接方式

Thrift	✓	1	ECU
Thrift2	✓	1	ECU
Rest Server	✓	1	ECU

Multi Tenant

- RegionServerGroup



HBase独立版

名称	资源分布情况	操作
ns0712	组：RegionGroup1(15 nodes) 流量：60MB/sec 请求数：10000 req/sec table数：10个 region数：100个	扩/缩容 查看历史 删除

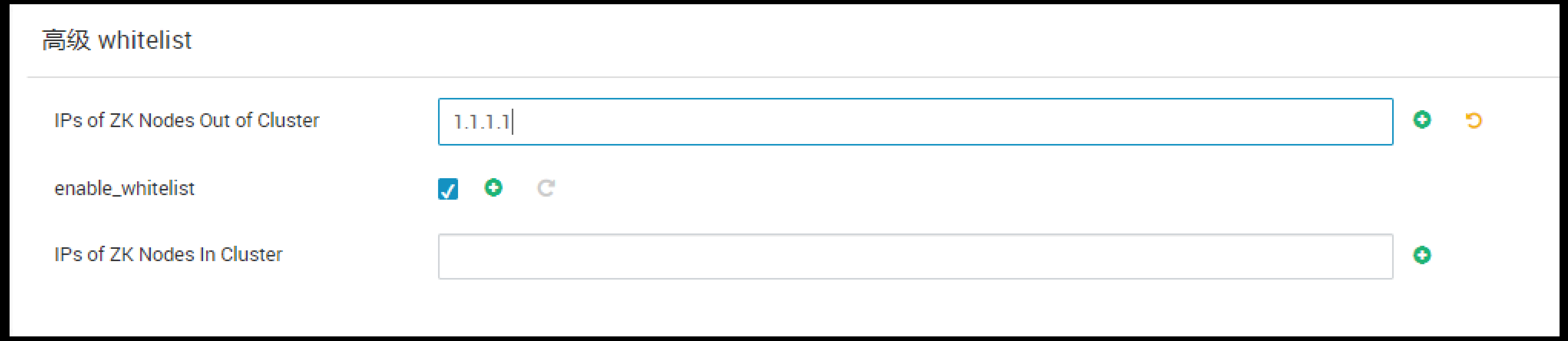
Multi Tenant

- qps and bandwith quota

```
set_quota TYPE => THROTTLE, USER => 'u1', LIMIT => '10M/sec'
set_quota TYPE => THROTTLE, USER => 'u1', LIMIT => ‘10000req/sec’
set_quota TYPE => THROTTLE, NAMESPACE => 'ns1', LIMIT => '10000req/sec'
```

Authorization & Authentication

1. Authorization
   1. auth base on Ranger
   2. orginal just support: admin/create/read/write
   3. more permission support :alter,drop,delete,modify,deletecolumn...
2. Authentication
   1. White List base on Zookeeper
   2. Use Kerberos or id/key authorization support
   3. BCID : id/key authorization ,simple and effective



高级 whitelist

IPs of ZK Nodes Out of Cluster	1.1.1.1
enable_whitelist	✓
IPs of ZK Nodes In Cluster	

Import config

RPC

handler **:** hbase.regionserver.handler.count

queue : separation of read/write

Read

1. BucketCache offheap
2. HDFS Short-Circuit Read
3. Hedged Read enable

Write

1. Bulkload
2. Multi wal
3. MSLAB ENABLE
4. blockingStoreFiles

Import config

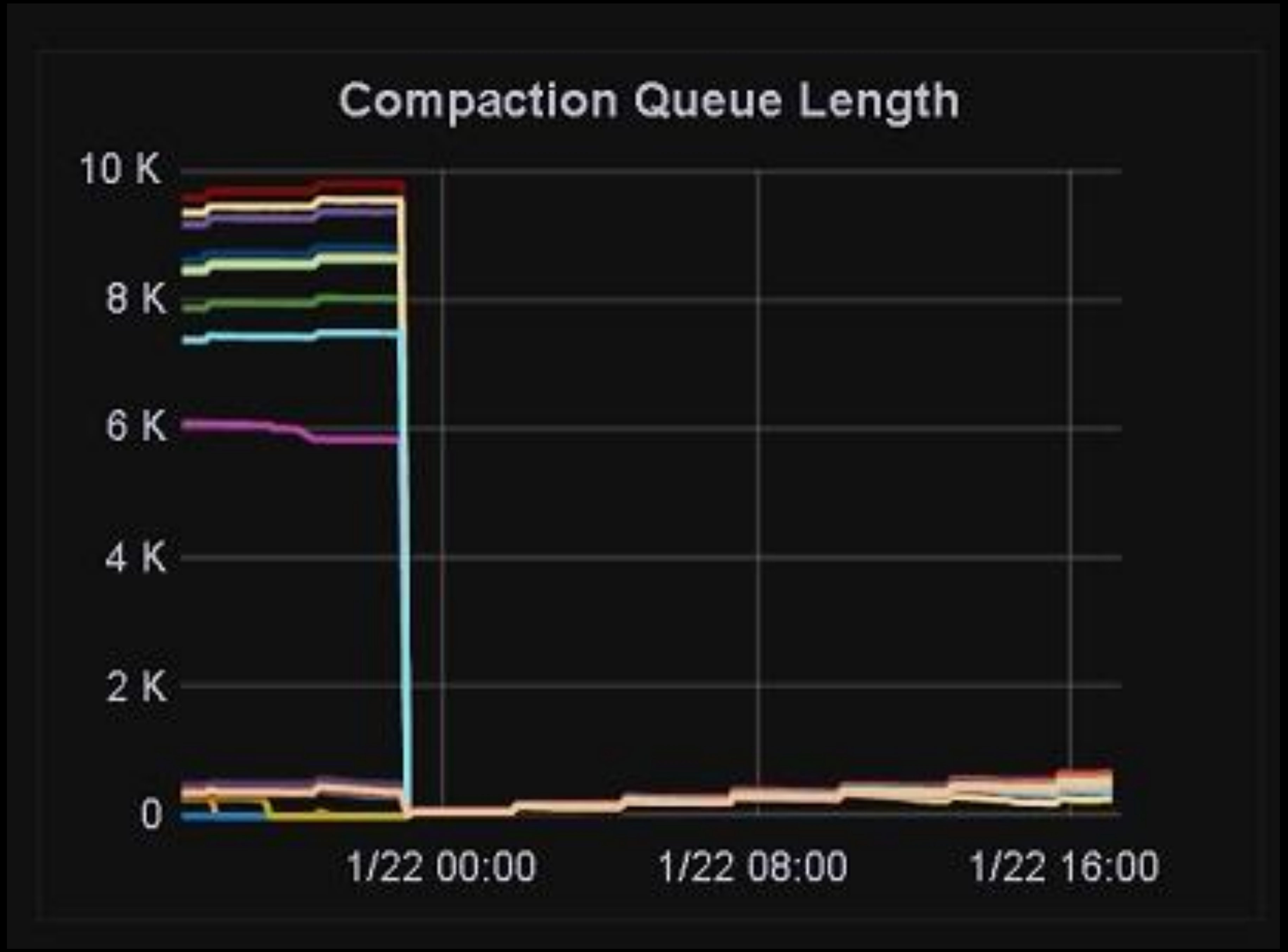
Compaction

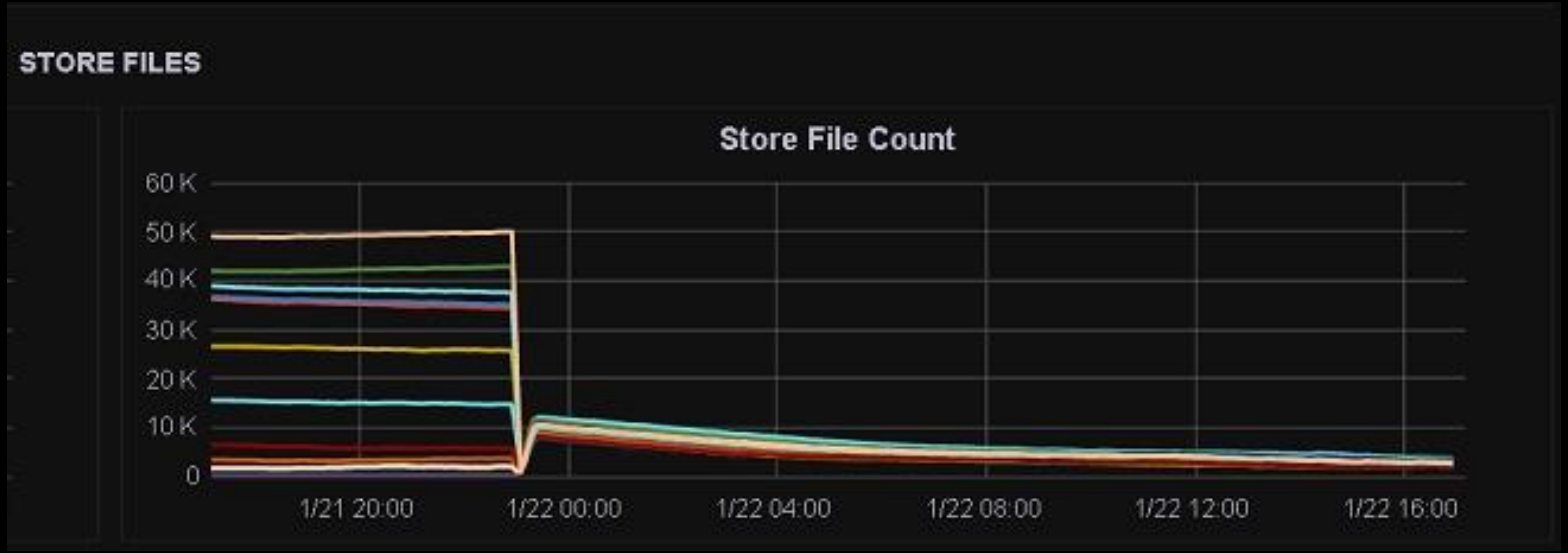
1. disable major compact, execute it offpeak time
   1. hbase.hregion.majorcompaction=0
2. Control the hfiles number : hbase.hstore.compaction.min/max 12/15
3. Control the hfiles number : hbase.hstore.compaction.max.size/min.size 5G/256M
4. set compaction threads : hbase.regionserver.thread.compaction.small/large 8/8

Loadbalance & SplitPolicy

1. SimpleLoadBalancer
2. ConstantSizeRegionSplitPolicy

Compaction tuning case





Import things-Schema Designs

1. Pre-split table
2. Rowkey design :Reversing/Hashing/Slating , such as reverse(phone_number)
3. SplitPolicy: ConstantSizeRegionSplitPolicy
4. Region Size : 10-50G
5. MAX_FILESIZE should larger than region size
6. Consider use Data Block Encoding when a row has many columns,but not use Prefix Tree
7. keep column family and qualifier short
8. Don’t put empty column

Import things-Schema Designs

Keep table size not too big still import

1. n+1 life data can fast drop table instead of compcation by TTL
2. Compaction can be faster : compaction just happened in current table , history is cold
3. Bulkload can be faster : one region one reduce,less regions means less reduce
4. Modify table can be faster : such as set compression gz and execute major_compact
5. RegionServer can handle more regions
6. Storage Policy can be used more flexible

Tools

1. Canary & hbck : check the rs/table/region status
2. gc log enable & /var/log/messages* : “Detected pause in JVM or host machine”
   1. **hard/soft lockup, CPU#16 stuck for 67s!**
   2. full gc
3. netstat/lsof : many tcp close_wait, such as HBASE-9393
4. jstack/jmap/gceasy… : why hbase stuck
5. Data migration : distcp+hbck /snapshot
6. Slow log: responseTooSlow/ TooLarge
7. Monitor
   1. HControl & Grafana & HMaster UI
   2. Regions/RPC/HFiles/Compaction/RIT/ProcessTime/Lantencies/Throughput/GC/Byte_in/out/ Locality/SlowOperation

04 Future

Future work

HBase on Colud

1. HBase service on China Mobile's Cloud
2. HBase on K8S
3. Separation of compute and storage
4. FileSystem with Cloud storage

HBase on modern hardware

1. SSD : Compaction policy base on SSD
2. Persistent Memory : wal /bucket cache,flush/bulkload HFile
3. RDMA : RoCE network support

Intelligent Operation and Maintenance System

Thanks！