### datanode\_startup\_procedures

#### 目录：

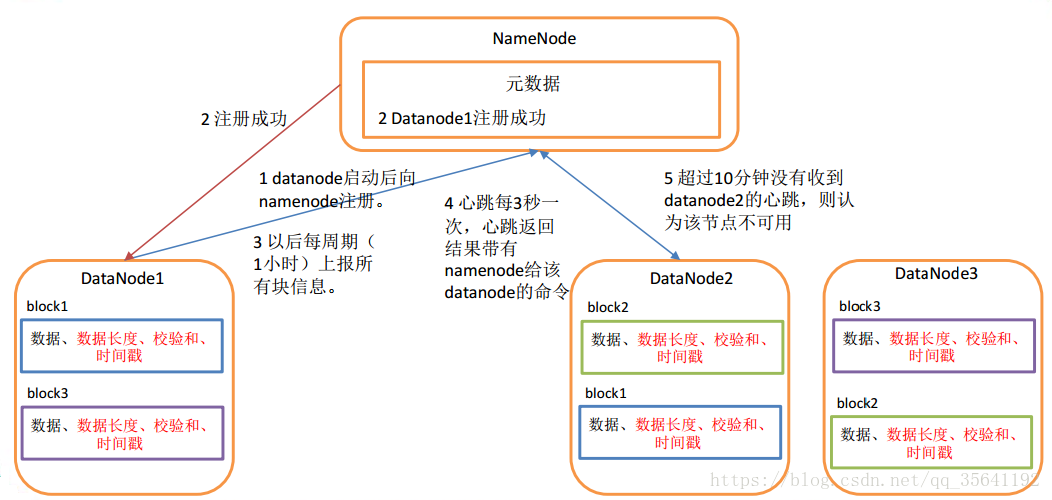
1. Datanode概述
2. Datanode启动流程
3. Datanode启动优化

#### 一．Datanode概述

**Datanode逻辑结构：**



**Datanode工作机制：**



**Datanode功能实现**

通过DataStorage以及FsDatasetImpl管理着数据节点存储上的所有数据块；

流式接口对客户端和其他数据节点提供读数据块、写数据块、复制数据块等功能；

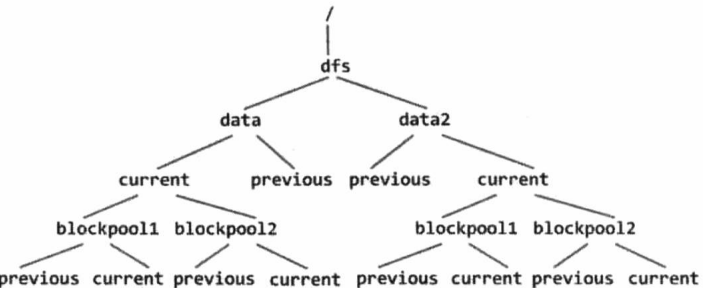
实现InterDatanodeProtocol以及ClientDatanodeProtocol，使得数据节点可以接收来自其他数据节点以及客户端的远程rpc请求；

BlockPoolManager对象周期性地向Namenode发送心跳、块汇报、增量块汇报以及缓存汇报，同时执行Namenode发回的command。

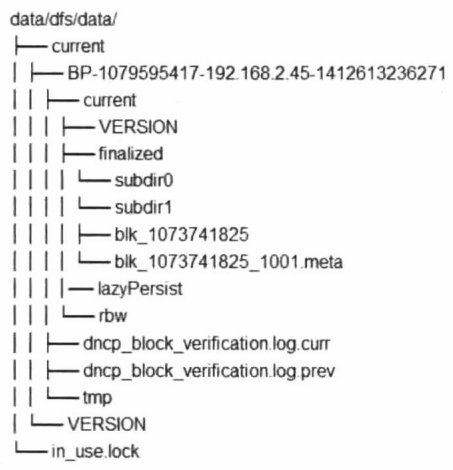
通过BlockScanner对象周期性地检查存储上的所有数据块。

DirectoryScanner对象验证存储上数据块和内存中数据块的一致性。

**Datanode磁盘存储结构**



存储目录、块池目录结构



Datanode磁盘存储结构

#### Datanode启动流程

**启动过程：**

加载配置文件进行初始化、磁盘检测；

启动多种工作线程，主要包括：

通信：BPServiceActor、IpcServer、DataXceiverServer、LocalDataXceiverServer

监控：BlockScanner、DirectoryScanner、JVMPauseMonitor

其他：InfoServer

向namenode注册，初始化存储结构：

包括各数据目录${dfs.datanode.data.dir}，及数据目录下各块池的存储结构

blockReport块汇报

**启动时间：**

集群版本： hadoop-2.7.2.3

启动耗时：60s （距离上一次停止时间未超过600s）

STARTUP\_MSG: 2018-12-06 15:38:54,267

Successfully sent block report ：2018-12-06 15:39:54,327

启动时间解析：

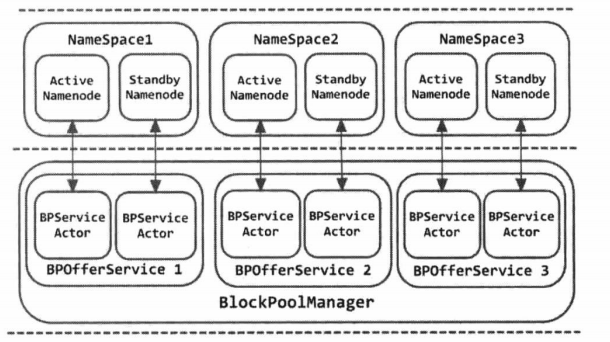
getAllVolumesMap ：Time to add replicas to map for block pool(12volumes)： 7079ms

BlockReport：Successfully sent block report 0x4b36a85614d760 ： 49s

**Datanode启动主流程：**



**BlockPoolManager**



**BPServiceActor**

主要功能：

与Namenode握手初始化命名空间对应块池的存储，注册当前datanode节点；

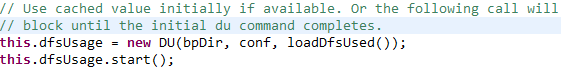
定期向Namenode发送心跳、块汇报、缓存汇报以及增量汇报；

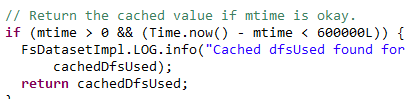
处理Namenode通过响应发回的名字节点指令；



**initBlockPool()**







**BlockReport：**

BlockReport :DataNode汇报所有数据块

对于节点规模较大和元数据量较大的集群，这个阶段的耗时会非常可观。主要有三点原因：

处理BlockReport的逻辑比较复杂，相对其他RPC操作耗时较长。

NameNode对每一个BlockReport的RPC请求处理都需要持有全局锁，也就是说对于BlockReport类型RPC请求实际上是串行处理；



**分析总结：**

datanode启动过程中会有多个线程，http和rpc的端口会先启动，但是这个时候还是不能提供服务的。当前版本（hadoop-2.7.2.3）主要耗时在构建volumeMap（需要遍历磁盘目录）与blockReport。

初始化完volumeMap会向namenode注册，注册成功后会在日志打印successfully registered with NN（有两条，active和standby namenode），之后datanode向namenode汇报块信息，发送完成后在日志打印Successfully sent block report（同样有两条）,只有在blockReport完成后才算真的启动完成。

集群节点比较多，NameNode重启时所有DataNode集中在同一时间段进行BlockReport请求。

#### Datanode启动流程优化

3.1 DataNode启动时getAllVolumesMap可以用文件缓存初始化，需要在DataNode关闭时把VolumesMap序列化到文件（HDFS-7928 ）

https://issues.apache.org/jira/browse/HDFS-7928

Scanning blocks from disk during rolling upgrade startup takes a lot of time if disks are busy

We observed this issue in rolling upgrade to 2.6.x on one of our cluster.   
One of the disks was very busy and it took long time to scan that disk compared to other disks.   
Seeing the sar (System Activity Reporter) data we saw that the particular disk was very busy performing IO operations.   
Requesting for an improvement during datanode rolling upgrade.   
During shutdown, we can persist the whole volume map on the disk and let the datanode read that file and create the volume map during startup after rolling upgrade.   
This will not require the datanode process to scan all the disk and read the block.   
This will significantly improve the datanode startup time.

3.2 DataNode ReplicaMap 锁使用（HDFS-10828 ）

https://issues.apache.org/jira/browse/HDFS-10828

Fix usage of FsDatasetImpl object lock in ReplicaMap

HDFS-10682 replaced the FsDatasetImpl object lock with a separate reentrant lock but missed updating an instance ReplicaMap still uses the FsDatasetImpl.

3.3 DataNode磁盘检测并行化改造

DataNode启动时，checkDiskError()检查并踢除坏掉的路径的地方可以并行检查并且不需要锁（HDFS-11086 )

https://issues.apache.org/jira/browse/HDFS-11086

DataNode disk check improvements

This Jira tracks a few improvements to DataNode’s usage of DiskChecker to address the following problems:

Checks are serialized so a single slow disk can indefinitely delay checking the rest.

Related to 1, no detection of stalled checks.

Lack of granularity. A single IO error initiates checking all disks.

Inconsistent activation. Some DataNode IO failures trigger disk checks but not all.

3.4 DataNode可优化内存占用来提高性能（HDFS-9260 HDFS-8859）

https://issues.apache.org/jira/browse/HDFS-9260

Improve the performance and GC friendliness of NameNode startup and full block reports

This patch changes the datastructures used for BlockInfos and Replicas to keep them sorted. This allows faster and more GC friendly handling of full block reports.

Would like to hear peoples feedback on this change.

https://issues.apache.org/jira/browse/HDFS-8859

Improve DataNode ReplicaMap memory footprint to save about 45%

3.5 DataNode启动时，初始化FsDatasetImpl锁粒度调整（HDFS-10682）

https://issues.apache.org/jira/browse/HDFS-10682

Replace FsDatasetImpl object lock with a separate lock object

This Jira proposes to replace the FsDatasetImpl object lock with a separate lock object. Doing so will make it easier to measure lock statistics like lock held time and warn about potential lock contention due to slow disk operations.   
Right now we can use org.apache.hadoop.util.AutoCloseableLock. In the future we can also consider replacing the lock with a read-write lock.

3.6 移除多余的Replica复制写操作

https://issues.apache.org/jira/browse/HDFS-8860

Remove unused Replica copyOnWrite code

3.7 通过NameNode的heartbeat消息进行DataNodes的Full BLock Reports流控（ HDFS-7923 ）

https://issues.apache.org/jira/browse/HDFS-7923

The DataNodes should rate-limit their full block reports by asking the NN on heartbeat messages (cmccabe)

The DataNodes should rate-limit their full block reports. They can do this by first sending a heartbeat message to the NN with an optional boolean set which requests permission to send a full block report. If the NN responds with another optional boolean set, the DN will send an FBR… if not, it will wait until later. This can be done compatibly with optional fields.