HDFS Checkpoint

Namenode元数据的持久化涉及到edits和fsimage两类数据。edits是HDFS操作的日志记录，在对HDFS进行修改操作之前，都会往edits中记录一条日志。FSimage存储HDFS中命名空间、数据块分布、文件属性等信息。edits在每次修改HDFS中都会插入记录，而fsimage在整个HDFS运行期间不会产生变化。

NameNode在启动时，从会把edits中的操作增加到fsimage中，并且把edits情况，所以fsimage总是记录启动namenode时的状态，而edits在每次启动时也是空的，只记录本次启动后的操作日志。Namenode启动时，edits的文件可能会增长到很大，需要花费很长时间来恢复，如果HDFS运行过程中发生Namenode故障，edits中的记录会丢失，因此需要利用Checkpoint将修改操作持久化。

# 1、Name启动时，FSimage与FSeditlog的merge (hadoop 1.0)

NameNode.main()是系统的入口，调用createNameNode(argv[],conf)创建Namenode实例。如果是FORMAT、FINALIZE等一些命令，调用对应的方法后退出，如果命令时Backup,CheckPoint及其他命令，通过new NameNode(conf) 创建实例。

Namenode(conf)调用initialize(conf)完成实例的初始化，读取配置文件、token配置。使用loadNameSystem(...)加载namenode持久化在硬盘的信息（初始化FSNameSystem），

并启动各个线程。

FSNameSystem的初始化，通过loadFSImage()来完成，首先获取fsimage文件，初始化FSnameSystem后，调用loadEdits(),完成edits和fsimage的merge.s

# 2、Checkpoint时，FSimage与FSeditlog的merge(hadoop 2.0)

在hadoop 2.0中增加了Checkpoint功能，在创建namenode的时候，并不会将editlog加载到namenode中。在HDFS中CheckPointer 触发周期性的checkpoint，主要的影响因素有时间和edit文件的大小。

在SecondaryNamenode.doWork()，根据配置获取checkpoint周期，如果now>

lastCheckpointTime+periodMSec，那么shouldCheckPoint = true，调用doCheckPoint()。

doCheckPoint():

1)从secondaryNamenode中的配置路径/tmp/hadoop/dfs/secondary下，获取editdir及checkpointDirs，并初始化checkpointImage（这步在secondarynamenode初始化过程中会初始化checkpointImage）

2) 获取RemoteEditLogManifest manifest，通过rpc调用获取远端namenode的editlog

3) 从namenode获取fsimage，TransferFSImage.downloadImageToStorage()

4) 调用doMerage(sig,manifest,loadImage,checkpointImage,namesyste)，完成fsimage和editlog的merge

5) 调用CheckPointer.roolForwardByApplyingLogs(manifest,dstimage,dstNs)

7) 将fsimage，发送到namenode当中

# 3、Checkpoint的导入

当Namenode发生故障丢失元数据后，使用SecondaryNameNode进行导入恢复:

在namenode的节点上创建dfs.name.dir目录，指定配置文件中的fs.checkpoint.dir

(hdfs-site.xml配置)，启动Namenode带上-importCheckPoint。Namenode将fs.checkpoint.dir中的文件拷贝到dfs.name.dir中。

启动namenode即可完成恢复