链滴

HDFS 底层读写原理

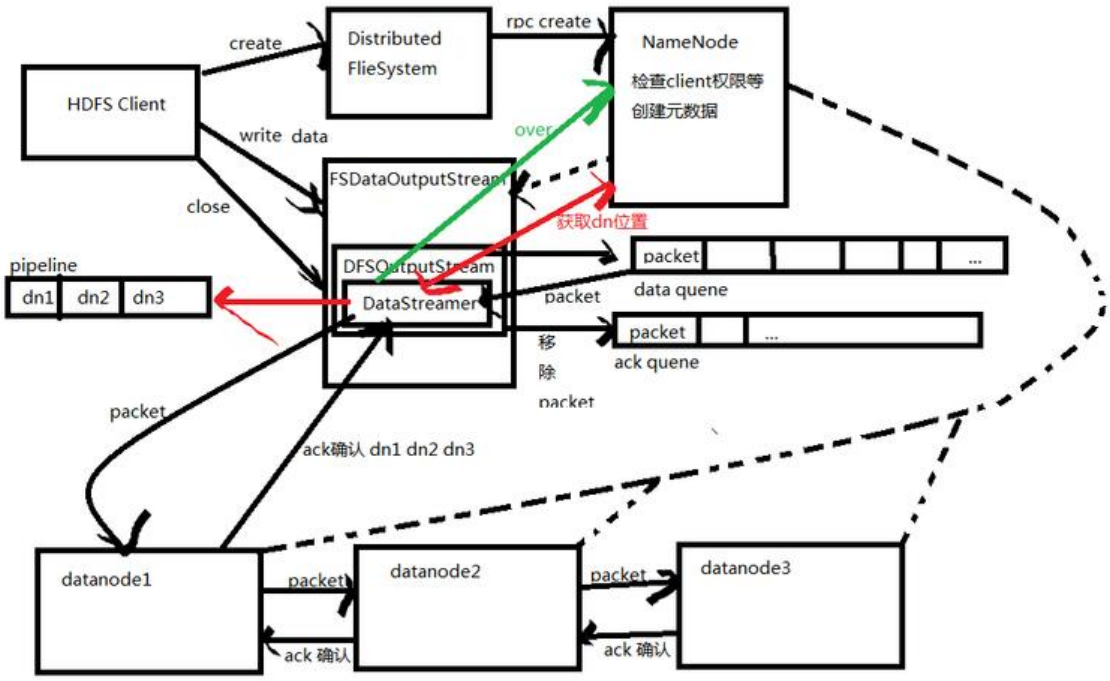
作者：stepforwards

原文链接：https://ld246.com/article/1570707340267

来源网站：链滴

hdfs底层读写原理

HDFS写示意图



写

```
Configuration conf = new Configuration();
FileSystem fs = FileSystem.get(conf);
```

原文链接：HDFS 底层读写原理

```
Path path = new Path("demo.txt");
FSDataOutputStream outStream = fs.create(path);
outStream.writeUTF("hello world");
outStream.flush();
outStream.close();
```

1.client 调用 DistributedFileSystem.create 创建新文件
2.DistributedFileSystem RPC 调用namenode去创建没有blocks关联的新文件
NameNode会做各种校验
文件是否存在，client是否有权限
返回FSDataOutputStream对象 本质是DFSOutputStream
client写数据到DFSOutputStream
DFSOutputStream 切割数据 packet
排成队列 data quene
3.DataStreamer 处理 data quene
询问 namenode 新的block 存储位置（datanode）
namenode 根据网络等因素 寻找最为合适的3（副本数）个 datanode
将datanode排成一个pipeline
将packet data quene输出到pipeline的第一个datanode，由第一个datanode将packet向pipelin中的第二个datanode传递，以此类推
datanode接收到packet后会向上一级反馈接受信息
4.DFSOutputStream 确认队列 ack quene 由 packet组成，
其接收到所有的pipeline中datanode的反馈信息后，会将ack quene 中相应的packet移除掉

如果在写的过程中pipeline中的某个datanode出错
删掉错误的datanode中未写完毕的block
剩下的两个正常的block继续执行packet的写入
namenode寻找新的datanode创建block的复制
5.client完毕写数据后调用close方法关闭写入流
6.DataStreamer收到最后一个ack后，通知datanode把文件视为写入完毕

读

```
FSDataInputStream inputStream = hdfs.open(path);
String content = inputStream.readUTF();
```

1.client open 调用DistributedFileSystem的实例的 open方法
2.DistributedFileSystem 通过rpc获得文件的第一批block的locations,同一个block根据副本数会返多个locations
这些locations依照hadoop拓扑结构排序，距离client近的排在前面
返回FSDataInputStream对象，该对象会被封装DFSInputStream对象，DFSInputStream能够方的管理datanode和namenode数据流
client调用read方法 DFSInputStream(根据网络等因素)找出离client近的datanode并连接
3.数据从datanode 流向client
4.当block读取完之后，就会关闭该block的datanode连接。接着读取下一块
5.全部的block读取完之后关闭所有的流

假设在读数据的时候，DFSInputStream和datanode的通讯发生异常，记录该datanode发生异常。试读locations中第二近的block，

DFSInputStream也会检查block数据校验和，假设发现一个坏的block,就会先报告到namenode节点然后DFSInputStream在其它的datanode上读该block的镜像。

```
//////////////////////////////
hdfs :// master:50070 /
bufferSize -> "io.file.buffer.size" default 4096
FileSystem.open(path f, int bufferSize)
DistributedFileSystem extends FileSystem Override open(path f,int bufferSize){
    增加一个读操作
    判断是否是绝对路径
    doCall
        DFSClient.open(pathsrc, bufferSize, verifyChecksum)
            new DFSInputStream(dfsClient,pathsrc,buffersize,verifyChecksum )
                dfsClient.getDefaultReadCachingStrategy()
                openInfo()
                    getblockinfo
        retrun FSDataInputStream(HdfsDataInputStream(DFSInputStream))
}

ha-hdfs:master
```