Computer Systems Engineering Lab3

Simple Distributed File System Part 3

Release: Nov 16, 2016

First Checkpoint: **Nov 23, 2016 24:00**

Deadline: **Nov 30, 2016 24:00**

Introduction

In this lab, you are required to deal with multi client problem and implement namenode log. Datanode log is not required.

Requirement

Your lab will be judged by following parts.

Thread-safe

During test, it is possible that multi client read and write the same file on the same time, thus it is necessary to ensure all functions are thread-safe.

Remind that your RPC server implement should use multi thread.

Also, we appreciate high performance. So the more concurrent the program is, the higher grade you will get.

For this part, if you is not thread-safe you will get 0/10. If you make all method synchronized, you will get 6/10. Thus, use less lock as possible while keep it thread-safe.

Multi client solution

You should consider how to solve read write conflict between multi client.

The following gif is a suggestion of procession: **详解"图解cow"目录下的gif**

My suggestion is use uuid to tag each file access. When client open a file, the name Node server should return a uuid to identification this access. And each access could be

readonly access or readwrite access. So there will be following situations:

> 1. Multi read access to the same file: It should always success.
> 2. Multi write access to the same file: Only the first write access success. The latter access should be denied.
> 3. Multi read access and one write access: Before write access close, all read access, including read request after write access begin, should refer to the version of file before modified. You should implement copy-on-write technique to ensure both read and write access will work correctly. When write access close, file metadata should be stored, but any read access should also refer the version of file before modified. Only when all read access of old version of file close could the name Node remove the old file metadata from its memory.

This part will be 35% at total

Log

You should implement log on namenode.

In this lab, you are suggested to implement following log:

> 1. Mkdir log
> 2. Create file log
> 3. Open readwrite file operation log
> 4. Add blocks log
> 5. New copy on write block log
> 6. Remove blocks log
> 7. Write commit log
> 8. Write abort log

Openreadonly log is not mandatory since it do not change any data.

Namenode should save entire file tree to disk at given period: flushDiskInternalSeconds. Before flush, all operation should only modify the memory version of the file tree, not on disk.

Please remember that log should be write to disk before the actual operation.

And when namenode init, it should read log and merge it into file tree.

Remind that namenode may be killed at any time, including while it is flushing data to disk. Any behaviors that may corrupt data are not allowed.

This part will be graded 25% at total.

Checkpoint submit

At checkpoint, you should implement thread-safe and Multi client solution. Thus you should pass all test case except LogTest.

Your document should including following part:

1. Your thread-safe design
2. Your multi client solution
3. Describe the problem you met during developing and how you solve it.
4. Briefly describe the change of given source code and test code(if possible)
5. Extra work you have done

Submit file tree:

> [YourStudentID] //the parent directory
> |-[YourStudentID]_Document-1.pdf
> |-src //the directory contains all your source code and test code

Please archive it as [YourStudentID]-1.zip or [YourStudentID]-1.tar.gz

Final submit

At this time, you should pass all test code. Notice that pass all test code does not promise you will get full grade.

Your document should including following part:

1. Your log design
2. How you process when you merge log and flush namenode data to disk.
3. Describe the problem you met during developing and how you solve it.
4. Briefly describe the change of given source code and test code(if possible)
5. Extra work you have done

Submit file tree:

> [YourStudentID] //the parent directory
> |-[YourStudentID]_Document-1.pdf |-[YourStudentID]_Document-2.pdf |-
> src //the directory contains all your source code and test code

Please archive it as [YourStudentID]-1.zip or [YourStudentID]-1.tar.gz

Grade

It is **your responsibility** to ensure all test case will be passed. Otherwise you will not get full grade.

It means that it is **your responsibility** to set up test case since it will not be correctly set up by default.

Feel free to modify any code given as long as it could pass the test case. DO NOT add or delete any line in the test case. However, you are allowed to modify the test case method call if it do not match your code.

Document: 30%, Code design 25%(15% Multi client solution + 10% Log), A correct program 40%(20% Multi client solution + 15% Log + 10% thread-safety), Bonus 20% **Any cheat behaviours(e.g. Copy code/design or code are too similar to any others/Internet) will get zero in all grading part**