# Project Appendix

Zhichao Yang, 20661179 & Liwen Dai, 20552153

## 1 Setups

Setup HDFS on pseudo distributed mode, with 2 replicas for each file, block size 64 MB:
hadoop-2.7.2/etc/hadoop/core-site.xml:

```
<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://localhost:9000</value>
    </property>
</configuration>
```

hadoop-2.7.2/etc/hadoop/hdfs-site.xml:

```
<configuration>
<property>
   <name>dfs.replication</name>
   <value>2</value>
 </property>
<property>
   <name>dfs.heartbeat.interval</name>
   <value>500</value>
 </property>
 <property>
   <name>dfs.namenode.name.dir</name>
   <value>file:/<path>/namenode</value>
 </property>
 <property>
   <name>dfs.datanode.data.dir</name>
   <value>file:/<path>/datanode</value>
 </property>
 <property>
    <name>dfs.block.size</name>
    <value>67108864</value>
    <description>Block size</description>
 </property>
</configuration>
```

Also need to configure ssh key to login localhost without password:
https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/SingleCluster.html#Pseudo-Distributed_Operation

Format the NameNode before launch:

```
$ hdfs namenode -format
```

Web interface for HDFS state http://localhost:50070/

To start HDFS,

```
$ cd hadoop-2.7.2/sbin
$ ./start-dfs.sh
$ jps
```

Now should see DataNode, NameNode, SecondaryNameNode and Jps.

To run additional data nodes,  use run-additionalDN.sh, modified from:
https://bigdata.wordpress.com/2010/05/27/hadoop-cookbook-4-how-to-run-multiple-data-nodes-on-one-machine/

```
#!/bin/sh
# This is used for starting multiple datanodes on the same machine.
# run it from hadoop-dir/ just like 'bin/hadoop'

#Usage: run-additionalDN.sh [start|stop] dnnumber
#e.g. run-datanode.sh start 2

DN_DIR_PREFIX="/home/<username>/mydata/hadoopTmp/dn"

if [ -z $DN_DIR_PREFIX ]; then
echo $0: DN_DIR_PREFIX is not set. set it to something like "/hadoopTmp/dn"
exit 1
fi

run_datanode () {
DN=$2
export HADOOP_LOG_DIR=$DN_DIR_PREFIX$DN/logs
export HADOOP_PID_DIR=$HADOOP_LOG_DIR
DN_CONF_OPTS="\
-Dhadoop.tmp.dir=$DN_DIR_PREFIX$DN"
sbin/hadoop-daemon.sh --script bin/hdfs $1 datanode $DN_CONF_OPTS
}

cmd=$1
shift;

for i in $*
do
run_datanode  $cmd $i
done
```

Before use modify the DN_DIR_PREFIX in run-additionalDN.sh to your path.

To run additional DataNodes, need multiple copies of hadoop directories.

```
$ cp -r hadoop-2.7.2/  node1
$ cp run-additionalDN.sh node1/
```

Modifiy node1/etc/hadoop/hdfs-site.xml  to use different directory and port numbers:

```
<configuration>
 <property>
   <name>dfs.datanode.data.dir</name>
   <value>file:/home/singulo/mydata/hdfs/datanode1</value>
 </property>
<property>
   <name>dfs.heartbeat.interval</name>
   <value>500</value>
 </property>
  <property>
   <name>dfs.datanode.address</name>
   <value>localhost:50110</value>
 </property>
    <property>
   <name>dfs.datanode.ipc.address</name>
   <value>localhost:50120</value>
 </property>
    <property>
   <name>dfs.datanode.http.address</name>
   <value>localhost:50130</value>
 </property>
</configuration>
```

```
$ cd node1
$ ./run-additionalDN.sh start 1
```

Now run $ jps again , should see an extra DataNode.

To run more data nodes, repeat the steps above, and don’t forget to use new name/directory/ports.

## 2. Tool Used:

We are using the unix tool ***strace*** to attain system calls on namenode and datanode. The tool strace provides various utilities that are very useful in getting time-based and count-based system call information to help us understand the file system access pattern in HDFS. The -p option in strace can attach the strace to the node process, and -f option can help trace all the child processes created by a node process. -c option can count all the system call upon the end of strace. -e option can help us filter the results we want to see.

Specifically, we have the following script to help automating our tests:

```
#!/bin/bash

rm -rf trace-hdfs-proc-out
mkdir trace-hdfs-proc-out

PASSWORD="your pass word"

jps | while read node ; do
    pid=$(echo $node | cut -d' ' -f1)
    nodeName=$(echo $node | cut -d' ' -f2)
    echo $PASSWORD | sudo -S strace $1 -t -e trace=file,desc -f -o
trace-hdfs-proc-out/${pid}.${nodeName}.txt -p $pid &
done

sleep 5
read  -n 1 -p "Press any key to stop:" key

echo $PASSWORD | sudo -S kill $(ps -aux | grep strace | tr -s ' ' | cut -d' ' -f2 )
```

## 3. Results:

### 3.1 File system access pattern when HDFS is idle:

Search system-wide file descriptor:

```
$  lsof | grep …
```

Or

```
$ readlink /proc/<pid>/fd/<fd number>
```

**3.1.1** Each of 3 DataNodes:
Repeatedly call epoll_wait(), on several file descriptors, where readlink returns anon_inode:[eventpoll] (what is this?)

Loop, heartbeat to NameNode:

    3 calls to statfs() on the root of node
    Try read() from TCP = -1 EAGAIN
    EPOLL_CTL_ADD
    write() to TCP fd, 398 bytes, (talk to NameNode)
    EPOLL_CTL_DEL
    Read 40 bytes from TCP
    Epoll_wait()

Every loop lasts for about 3 seconds.
Notices that there’s no writes to the disc.
The default heartbeat time is 3 seconds, change it to 500 in the configuration file to remove the “heartbeat noise” in the result. In every hdfs-site.xml file, add:

```
 <property>
  <name>dfs.heartbeat.interval</name>
  <value>500</value>
  <description>Heatbeat interval</description>
</property>
```

**3.1.2** NameNode:
Loop:

    statfs() on the node root
    Epoll_wait
    From each DataNode, read 4 + 394 = 398 bytes from a TCP socket, the reply 40 bytes

Each loop lasts about 5 seconds.
Besides, Fcntl F_GETFL, F_GETFL appeared 3 times

**3.1.3**Secondary NameNode:
Log of trace is only 1-page long, mostly epoll_wait(), with few small reads/writes, epoll_ctl, fcntl, dup2, mmap(size=12288)

Understanding what HDFS is doing at idle state helps to remove “noise” at real workload analysis.

**3.2 File system access pattern when appending short text to a file:**

We keep appending a text file of 5.1 MB, Shakespeare.txt, to the HDFS.

```
#!/bin/bash
for i in $(seq 1 50) ; do hdfs dfs -appendToFile Shakespeare.txt Shakespeare-copy.txt ; done
```

**3.2.1** DataNode:
Before writing the actual content, a datanode checks stats of, and reads from many java class, jar and xml files, and it keeps calling lseek() and read() on hadoop-hdfs-2.7.2.jar (fd = 107), where most reads are 30 bytes

```
25644 19:22:32
stat("/home/singulo/hadoop-2.7.2/etc/hadoop/org/apache/hadoop/hdfs/protocol/proto/HdfsProtos$B
lockProto.class", 0x7fad8be3f200) = -1 ENOENT (No such file or directory)
25644 19:22:32
stat("/home/singulo/hadoop-2.7.2/share/hadoop/hdfs/org/apache/hadoop/hdfs/protocol/proto/HdfsP
rotos$BlockProto.class", 0x7fad8be3f200) = -1 ENOENT (No such file or directory)
25644 19:22:32 lseek(107, 469827, SEEK_SET) = 469827
25644 19:22:32 read(107, "PK\3\4\24\0\10\0\10\0Z\1:H\0\0\0\0\0\0\0\0\0\0\0A\0\0\0", 30) = 30
25644 19:22:32 lseek(107, 469922, SEEK_SET) = 469922
25644 19:22:32 read(107, "\275Z\vx\24\327u\376\357jwGZFBO@,
\f\2\213\225@\17\236F26BB,"..., 4463) = 4463
25644 19:22:32 mmap(0x7fad8a3ee000, 262144, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fad8a3ee000
```

During this process, the block files blk_xxx and blk_xxx.meta are created under BP-xxx/current/rbw/ directory, which represents the “Replica Being Written” state.

```
2893  19:22:32
stat("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/curr
ent/rbw/blk_1073741827", 0x7fad8a62d200) = -1 ENOENT (No such file or directory)
2893  19:22:32
open("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/cur
rent/rbw/blk_1073741827", O_RDWR|O_CREAT|O_EXCL, 0666) = 324
2893  19:22:32 fstat(324, {st_mode=S_IFREG|0664, st_size=0, ...}) = 0
2893  19:22:32 close(324)
…
2893  19:22:32
open("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/cur
rent/rbw/blk_1073741827_1042.meta", O_RDWR|O_CREAT, 0666) = 324
2893  19:22:32 fstat(324, {st_mode=S_IFREG|0664, st_size=0, ...}) = 0
2893  19:22:32
open("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/cur
rent/rbw/blk_1073741827", O_RDWR|O_CREAT, 0666) = 325
2893  19:22:32 fstat(325, {st_mode=S_IFREG|0664, st_size=0, ...}) = 0
```

64 KBytes at a time, the datanode reads the data from a socket (fd = 316), and writes to another socket (fd=326), to another replica. Then it appends the data to the blk file, and appends 504 bytes to the blk.meta file.

```
2893  19:22:32 read(316,
"\0\0\375\374\0\31\t\0\340\7\0\0\0\0\0\21\10\0\0\0\0\0\0\0\30\0%\0\374\0\0\305"..., 512) = 512
2893  19:22:32 read(316, "5'Y&9\222$\352\3138\25\373\277\335\2\251b\357\257,b\33\367 of
fores"..., 64535) = 64535
2893  19:22:32 write(326,
"\0\0\375\374\0\31\t\0\340\7\0\0\0\0\0\21\10\0\0\0\0\0\0\0\30\0%\0\374\0\0\305"..., 65047) = 65047
2893  19:22:32 write(325, " of foresters\n\n  JAQUES. Which i"..., 64512) = 64512
2893  19:22:32 write(324,
"\305\4W\375\0374\327\10k\275u\1.\236\326^\326\30\3\377\310\271;\257\4\1\332\260\331\201\1#"
..., 504) = 504
...
```

When the actual data have been written, the datanode rename the rbw/blk and rbw/blk_.meta files to the finalized/ directory. Then it writes to its own log file  (357 bytes), and acknowledges back through the socket.

```
2896  19:22:32
stat("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/curr
ent/finalized/subdir0/subdir0", {st_mode=S_IFDIR|0775, st_size=4096, ...}) = 0
2896  19:22:32
rename("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/
current/rbw/blk_1073741827_1042.meta",
"/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/current/fi
nalized/subdir0/subdir0/blk_1073741827_1042.meta") = 0
2896  19:22:32
rename("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/
current/rbw/blk_1073741827",
"/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/current/fi
nalized/subdir0/subdir0/blk_1073741827") = 0
2896  19:22:32
```

```
stat("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/curr
ent/finalized/subdir0/subdir0/blk_1073741827_1042.meta", {st_mode=S_IFREG|0664,
st_size=41635, ...}) = 0
...
2896  19:22:32 write(191, "2017-03-29 19:22:32,925 INFO org"..., 357 <unfinished ...>
...
2896  19:22:32 <... write resumed> )    = 357
2896  19:22:32 write(316, "\20\10\246\1\20\0\20\0\30\277\324\252\1\"\2\0\0", 17 <unfinished ...>
...
2896  19:22:32 <... write resumed> )    = 17
```

After that, the datanode calls close() to more than 300 file descriptors:

```
...
2986  19:22:34 close(319)             = 0
2986  19:22:34 close(320)             = 0
2986  19:22:34 close(321)             = 0
2986  19:22:34 close(322)             = 0
2986  19:22:34 close(323)             = 0
2986  19:22:34 close(327)             = 0
2986  19:22:34 close(328)             = 0
2986  19:22:34 close(329)             = 0
```

When the next append command comes, finalized/blk_xxx(.meta) are renamed back to rbw/ directory, and the pattern above repeats.

```
2985  19:22:34 statfs("/home/singulo/mydata/hdfs/datanode", {f_type="EXT2_SUPER_MAGIC",
f_bsize=4096, f_blocks=27892164, f_bfree=17804223, f_bavail=16381619, f_files=7094272,
f_ffree=6501723, f_fsid={-1149700033, -846037428}, f_namelen=255, f_frsize=4096,
f_flags=4128}) = 0
2985  19:22:34 statfs("/home/singulo/mydata/hdfs/datanode", {f_type="EXT2_SUPER_MAGIC",
f_bsize=4096, f_blocks=27892164, f_bfree=17804223, f_bavail=16381619, f_files=7094272,
f_ffree=6501723, f_fsid={-1149700033, -846037428}, f_namelen=255, f_frsize=4096,
f_flags=4128}) = 0
2985  19:22:34
rename("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/
current/finalized/subdir0/subdir0/blk_1073741827_1042.meta",
"/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/current/r
bw/blk_1073741827_1043.meta") = 0
2985  19:22:34
rename("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/
current/finalized/subdir0/subdir0/blk_1073741827",
"/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/current/r
bw/blk_1073741827") = 0
2985  19:22:34
open("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/cur
rent/rbw/blk_1073741827_1043.meta", O_RDWR|O_CREAT, 0666) = 324
2985  19:22:34 fstat(324, {st_mode=S_IFREG|0664, st_size=41635, ...}) = 0
….
```

When the data file size exceeds 64 MB, the block size we set, the new rbw/blk files are created accordingly.

Call Summary from a DataNode:

```
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 99.52    0.496050         432      1148           epoll_wait
  0.43    0.002119           1      4043        28 write
  0.05    0.000258           0      4159       445 read
  0.00    0.000018           0       666           mmap
  0.00    0.000012           0       968           epoll_ctl
  0.00    0.000000           0      1107       792 open
  0.00    0.000000           0     10355           close
  0.00    0.000000           0       203        88 stat
  0.00    0.000000           0       495           fstat
  0.00    0.000000           0       220           lstat
  0.00    0.000000           0        98           lseek
  0.00    0.000000           0       144           ioctl
  0.00    0.000000           0       199       199 access
  0.00    0.000000           0       147           pipe
  0.00    0.000000           0       141           dup2
  0.00    0.000000           0       232       203 execve
  0.00    0.000000           0       191           fcntl
  0.00    0.000000           0        72           getdents
  0.00    0.000000           0        58           rename
  0.00    0.000000           0        95        56 statfs
  0.00    0.000000           0         2           epoll_create
  0.00    0.000000           0         7           openat
  0.00    0.000000           0        31           newfstatat
------ ----------- ----------- --------- --------- ----------------
100.00    0.498457                 24781      1811 total
```

**3.2.2** NameNode:
The NameNode writes to its own log (fd = 191), with about 200 bytes each time, and talks through socket (fd = 223).

```
25465 19:23:12 write(223,
"\0\0\0005\32\10\3\20\0\30\t:\20\272\367\331\34\334\236LP\241N\326\36\26\4\202i@\0\31"..., 57)
= 57
25452 19:23:12 <... epoll_wait resumed> [{EPOLLIN, {u32=223, u64=8340593504036061407}}],
8192, -1) = 1
25452 19:23:12 read(223, "\0\0\1\364", 4) = 4
25452 19:23:12 read(223,
"\32\10\2\20\0\30\10\"\20\272\367\331\34\334\236LP\241N\326\36\26\4\202i(\0B\n\16up"..., 500) =
500
25452 19:23:12 epoll_wait(215,  <unfinished ...>
25473 19:23:12 write(191, "2017-03-29 19:23:12,826 INFO org"..., 240) = 240
25473 19:23:12 write(191, "2017-03-29 19:23:12,826 INFO Blo"..., 161) = 161
25473 19:23:12 write(191, "2017-03-29 19:23:12,826 INFO Blo"..., 161) = 161
```

Similar to DataNode, the NameNode also performs lots reads and stats checks on various code files (.class, libnio.so, etc…). However, no rename() is observed, and the actual data doesn't go though the NameNode.

The NameNode call summary:

```
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 96.40    0.431255        583       740          epoll_wait
  2.68    0.012000         59       204         fdatasync
  0.92    0.004097          4      1047         write
  0.00    0.000000          0      1245        57 read
  0.00    0.000000          0        11          open
  0.00    0.000000          0        67          close
  0.00    0.000000          0       216          fstat
  0.00    0.000000          0       226          lseek
  0.00    0.000000          0        27          mmap
  0.00    0.000000          0        55          dup2
  0.00    0.000000          0       171          fcntl
  0.00    0.000000          0        18          statfs
  0.00    0.000000          0       112        55 epoll_ctl
------ ----------- ----------- --------- --------- ----------------
100.00    0.447352                  4139       112 total
```

**3.3 File system access pattern for large writes:**

We try to put out.actor-collaboration, a text file of 452 MB, to HDFS twice:

```
#!/bin/bash
hdfs dfs -put out.actor-collaboration out.actor-collaboration-1
hdfs dfs -put out.actor-collaboration out.actor-collaboration-2
```

The NameNode turns out pretty much similar to the case of short writes.

```
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 86.38    0.076243        641       119          epoll_wait
 13.60    0.012000        600        20          fdatasync
  0.02    0.000017          1        18          fcntl
  0.00    0.000000          0       165         6 read
  0.00    0.000000          0       132          write
  0.00    0.000000          0         5          close
  0.00    0.000000          0        20          fstat
  0.00    0.000000          0        20          lseek
  0.00    0.000000          0         5          dup2
  0.00    0.000000          0         8          statfs
  0.00    0.000000          0        11         5 epoll_ctl
------ ----------- ----------- --------- --------- ----------------
100.00    0.088260                   523        11 total
```

DataNodes are also similar to the case of short writes, except that the read/write operations are much more intensive.

```
16113 23:19:44 statfs("/home/singulo/mydata/hdfs/datanode1", {f_type="EXT2_SUPER_MAGIC",
f_bsize=4096, f_blocks=27892164, f_bfree=17413536, f_bavail=15990932, f_files=7094272,
f_ffree=6496257, f_fsid={-1149700033, -846037428}, f_namelen=255, f_frsize=4096,
f_flags=4128}) = 0
16113 23:19:44 statfs("/home/singulo/mydata/hdfs/datanode1", {f_type="EXT2_SUPER_MAGIC",
f_bsize=4096, f_blocks=27892164, f_bfree=17413536, f_bavail=15990932, f_files=7094272,
f_ffree=6496257, f_fsid={-1149700033, -846037428}, f_namelen=255, f_frsize=4096,
f_flags=4128}) = 0
16113 23:19:44
stat("/home/singulo/mydata/hdfs/datanode1/current/BP-1586938737-127.0.1.1-1490573390385/cur
rent/rbw/blk_1073741839", 0x7f2d779a8200) = -1 ENOENT (No such file or directory)
16113 23:19:44
open("/home/singulo/mydata/hdfs/datanode1/current/BP-1586938737-127.0.1.1-1490573390385/c
urrent/rbw/blk_1073741839", O_RDWR|O_CREAT|O_EXCL, 0666) = 319
16113 23:19:44 fstat(319, {st_mode=S_IFREG|0664, st_size=0, ...}) = 0
16113 23:19:44 close(319)              = 0
16113 23:19:44
open("/home/singulo/mydata/hdfs/datanode1/current/BP-1586938737-127.0.1.1-1490573390385/c
urrent/rbw/blk_1073741839_1103.meta", O_RDWR|O_CREAT, 0666) = 319
16113 23:19:44 fstat(319, {st_mode=S_IFREG|0664, st_size=0, ...}) = 0
16113 23:19:44
open("/home/singulo/mydata/hdfs/datanode1/current/BP-1586938737-127.0.1.1-1490573390385/c
urrent/rbw/blk_1073741839", O_RDWR|O_CREAT, 0666) = 320
...

16113 23:19:44 read(317,
"X\7\246\236\17\355\32N\226\242\266[4\rp8;FNV?1:\225\275\315\327\314L\367mc"..., 64570) =
64570
16113 23:19:44 write(326,
"\0\0\375\374\0\31\t\0\2\0\0\0\0\0\0\21\24\4\0\0\0\0\0\0\30\0%\0\374\0\0\235"..., 65047) = 65047
16113 23:19:44 write(320, "\n11419 8415 \n11419 8896 \n11419 1"..., 64512) = 64512
16113 23:19:44 write(319,
"\235!h\211\230\267\242]\200\250\311]\25\362F\232^\4\313|\357\335\272\"\336\226\304C\251\\\20
45"..., 504) = 504
16113 23:19:44 read(317,
"\0\0\375\374\0\31\t\0\376\0\0\0\0\0\0\21\25\4\0\0\0\0\0\0\30\0%\0\374\0\0\25"..., 512) = 512
16113 23:19:44 read(317, "'h\335\3111\310\356\4J\303\0302+\24YL|\231\301\362\327\224#752
11661"..., 64535) = 64535
...
```

```
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 99.73    1.598498         380      4204           epoll_wait
  0.20    0.003261           0     19441        57 write
  0.06    0.001041           0     17612      2007 read
  0.00    0.000072           0      4135           epoll_ctl
  0.00    0.000000           0        18           open
  0.00    0.000000           0        30           close
  0.00    0.000000           0        18         6 stat
  0.00    0.000000           0        18           fstat
  0.00    0.000000           0        27           mmap
  0.00    0.000000           0         1           pipe
  0.00    0.000000           0         9           dup2
  0.00    0.000000           0        40           fcntl
  0.00    0.000000           0        12           rename
  0.00    0.000000           0        15           statfs
  0.00    0.000000          0         1          epoll_create
------ ----------- ----------- --------- --------- ----------------
100.00    1.602872              45581      2070 total
```

Note: epoll_wait() happens when read from a socket returns an error, for example:

```
15983 23:19:34 read(318, 0x7fad98221820, 512) = -1 EAGAIN (Resource temporarily unavailable)
15983 23:19:34 epoll_ctl(329, EPOLL_CTL_ADD, 318, {EPOLLIN, {u32=318, u64=4294967614}})
= 0
15983 23:19:34 epoll_wait(329, [{EPOLLIN, {u32=318, u64=4294967614}}], 8192, 60000) = 1
15983 23:19:34 epoll_ctl(329, EPOLL_CTL_DEL, 318, 0x7fad8a62d2e0) = 0
15983 23:19:34 epoll_wait(329, [], 8192, 0) = 0
15983 23:19:34 read(318, "\0\34P\252\1\ni\nA\n5\n%BP-1586938737-127.0"..., 512) = 175
```

**3. 4 File system access pattern for reads:**

```
#!/bin/bash

hdfs dfs -cat out.actor-collaboration-1 > /dev/null
hdfs dfs -cat out.actor-collaboration-2 > /dev/null
```

**3.4.1** NameNode: most are short reads/writes to the sockets.  No writes to local file system are observed.
Reads lib/guava-11.0.2.jar etc…

```
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
100.00    0.036000       1161       31          epoll_wait
  0.00    0.000000          0       42        4 read
  0.00    0.000000          0       12          write
  0.00    0.000000          0        2          close
  0.00    0.000000          0        1          mmap
  0.00    0.000000          0        2          dup2
  0.00    0.000000          0       12          fcntl
  0.00    0.000000          0        2          statfs
  0.00    0.000000          0        6        2 epoll_ctl
------ ----------- ----------- --------- --------- ----------------
100.00    0.036000                 110         6 total
```

**3.4.2** DataNode:

```
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 99.92    0.280019         50     5632          epoll_wait
  0.05    0.000132          0     2801          sendfile
  0.02    0.000066          0     5614          epoll_ctl
  0.01    0.000022          0     2809          write
  0.00    0.000008          0      367        5 read
  0.00    0.000006          0     3157          fstat
  0.00    0.000000          0        6          open
  0.00    0.000000          0         7          close
  0.00    0.000000          0         9          stat
  0.00    0.000000          0       353           lseek
  0.00    0.000000          0         3          mmap
  0.00    0.000000          0         1          dup2
  0.00    0.000000          0         9          fcntl
  0.00    0.000000          0         3          statfs
  0.00    0.000000          0        91           fadvise64
------ ----------- ----------- --------- --------- ----------------
100.00    0.280253               20862         5 total
```

For each block, only 1 replica is read. First, the block and the meta files are opened:

```
31179 16:04:22 read(316, "\0\34Qt\ng\nA\n5\n%BP-1586938737-127.0."..., 512) = 120
31179 16:04:22
stat("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/curr
ent/finalized/subdir0/subdir0/blk_1073741850", {st_mode=S_IFREG|0664, st_size=67108864, ...})
= 0
31179 16:04:22
stat("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/curr
ent/finalized/subdir0/subdir0/blk_1073741850_1114.meta", {st_mode=S_IFREG|0664,
st_size=524295, ...}) = 0
31179 16:04:22
open("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/cur
rent/finalized/subdir0/subdir0/blk_1073741850_1114.meta", O_RDONLY) = 318
31179 16:04:22 fstat(318, {st_mode=S_IFREG|0664, st_size=524295, ...}) = 0
31179 16:04:22
stat("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/curr
ent/finalized/subdir0/subdir0/blk_1073741850_1114.meta", {st_mode=S_IFREG|0664,
st_size=524295, ...}) = 0
31179 16:04:22 read(318, "\0\1\2\0\0\2\0\324\35\252wt\372\325#
[\335\"y'\3\305\274c>\323\242\353\t\334\24"..., 4096) = 4096
31179 16:04:22
open("/home/singulo/mydata/hdfs/datanode/current/BP-1586938737-127.0.1.1-1490573390385/cur
rent/finalized/subdir0/subdir0/blk_1073741850", O_RDONLY) = 319
31179 16:04:22 fstat(319, {st_mode=S_IFREG|0664, st_size=67108864, ...}) = 0
```

File content is sent by sendfile(), 64 KB at a time. At the same time, meta-data of 543 bytes are transferred through write() operation. Note that epoll_wait() is called to wait the socket to be available for writing.

```
31179 16:04:22 epoll_ctl(328, EPOLL_CTL_ADD, 316, {EPOLLOUT, {u32=316,
u64=16965504754916524348}}) = 0
31179 16:04:22 epoll_wait(328, [{EPOLLOUT, {u32=316, u64=16965504754916524348}}], 8192,
480000) = 1
31179 16:04:22 epoll_ctl(328, EPOLL_CTL_DEL, 316, 0x7f965d090180) = 0
31179 16:04:22 epoll_wait(328, [], 8192, 0) = 0
31179 16:04:22 fstat(319, {st_mode=S_IFREG|0664, st_size=67108864, ...}) = 0
31179 16:04:22 sendfile(316, 319, [0] => [65536], 65536) = 65536
31179 16:04:22 write(316, "\0\1\2\4\0\31\t\0\0\1\0\0\0\0\0\21\1\0\0\0\0\0\0\30\0%\0\0\1\0\\"..., 543)
= 543
...
```

**3.5 File system access pattern when creating small files:**

```
#!/bin/bash
for num in {1..100}
do
        echo 'hello world' | hdfs dfs -put - /test/$num.txt
done
```

The test try to create 100 new files containing “hello world” under the folder /test.

1) NameNode:
    a) Most frequently used system calls

```
Strace with -c option result:
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 86.21    0.025465          64       400           fdatasync
 13.54    0.004000           8       527           lseek
  0.19    0.000055           0      1728           write
  0.03    0.000009           0       416           fstat
  0.03    0.000008           0      1892           mprotect
  0.00    0.000000           0      2697       108 read
  0.00    0.000000           0        15           open
  0.00    0.000000           0       121           close
  0.00    0.000000           0        43        43 stat
  0.00    0.000000           0        55           mmap
  0.00    0.000000           0        11           munmap
  0.00    0.000000           0         1           rt_sigreturn
  0.00    0.000000           0        25           sched_yield
  0.00    0.000000           0       106           dup2
  0.00    0.000000           0       216       108 accept
  0.00    0.000000           0       106           shutdown
  0.00    0.000000           0       216           getsockname
  0.00    0.000000           0       216           setsockopt
  0.00    0.000000           0       324           fcntl
  0.00    0.000000           0        79           statfs
  0.00    0.000000           0       214       106 epoll_ctl
------ ----------- ----------- --------- --------- ----------------
100.00    0.029537                  9408       365 total
```

According to the above statistic, The most frequently used system calls related to file system are 2697 read(), 1728 write(), 527 lseek(), 400 fdatasync().

b) File system access pattern:

```
4621  22:48:56 lseek(208, 0, SEEK_CUR)  = 101534
4621  22:48:56 fstat(208, {st_mode=S_IFREG|0664, st_size=1048576, ...}) = 0
4621  22:48:56 write(208,
"\0\0\0\0\253\0\0\0\0\0\0\4\323\0\0\0\0\0\0@\323\0\25/test/2.txt._COPYING_\0\1\0\0\1[\27\367Uq\0\0\1[\27\367Uq\0\0\0\0\10\0\0\0\0\0\0\0\10holmesin\nsupergroup\1\244\0\0\0\0"..., 176) = 176
4621  22:48:56 fdatasync(208)          = 0
……
4627  22:48:56 lseek(208, 0, SEEK_CUR)  = 101710
4627  22:48:56 fstat(208, {st_mode=S_IFREG|0664, st_size=1048576, ...}) = 0
4627  22:48:56 write(208, "
\0\0\0\24\0\0\0\0\0\0\4\324\0\0\0\0@\0\0\314\373S8\10\37\0\0\0\24\0\0\0\0\0\0\4\325\0\0\0\0\0\0\4\2
64\335\267\360f!\0\0\0006\0\0\0\0\0\0\4\326\0\25/test/2.txt._COPYING_\1\0\0\0\0@\0\0\314\0\216\
4\264\0"..., 109) = 109
4627  22:48:56 fdatasync(208)          = 0
……
4624  22:48:57 lseek(208, 0, SEEK_CUR)  = 101819
4624  22:48:57 fstat(208, {st_mode=S_IFREG|0664, st_size=1048576, ...}) = 0
4624  22:48:57 write(208,
"\t\0\0\0w\0\0\0\0\0\0\4\327\0\0\0\0\0\0\0\0\0\25/test/2.txt._COPYING_\0\1\0\0\1[\27\367V\361\0\0\1
[\27\367Uq\0\0\0\0\10\0\0\0\0\0\0\1\0\0\0\0@\0\0\314\0\0\0\0\0\0\0\f\0\0\0\0\0\0\4\264\10h"..., 124)
= 124
4624  22:48:57 fdatasync(208)          = 0
……
4623  22:48:57 lseek(208, 0, SEEK_CUR)  = 101943
4623  22:48:57 fstat(208, {st_mode=S_IFREG|0664, st_size=1048576, ...}) = 0
4623  22:48:57 write(208,
"\1\0\0\0N\0\0\0\0\0\0\4\330\0\25/test/2.txt._COPYING_\0\v/test/2.txt\0\0\1[\27\367V\370\0\20U\177
\f\17\25\305A'\251Y;\322*|pQ\0\0\0\7f\275\2128", 83) = 83
4623  22:48:57 fdatasync(208)          = 0
```

After creation of the file on the datanode, the namenode will append the metadata to the file with the file handler 208, the file created has a suffix “_COPYING_”. This metadata is flushed to disk. Then after block added on the datanode, metadata will be appended to file 208, and flushed to disk. After the completion of the file creation, metadata will be flushed to disk for file 208. Then file will be renamed without the “_COPYING_” suffix, this metadata is write to file 208, and flushed to disk. To conclude, for each creation of a small file, it involves 4 metadata write and fdatasync() operation.

c) Finding:

After three times run of the 100 creation process, some different file access pattern was found.

```
4620  22:51:01 write(191, "2017-03-28 22:51:01,667 INFO
org.apache.hadoop.hdfs.server.namenode.FSEditLog: Starting log segment "..., 108) = 108
4620  22:51:01
open("/tmp/hadoop-holmesin/dfs/name/current/edits_inprogress_0000000000000001500",
O_RDWR|O_CREAT, 0666) = 208
```

During the operation, some operation relates to rolling edit logs happens. And the file above was created with file handle 208. After observation, all the new metadata change will try to write to this file. The log information in the above table implies the metadata on the namenode is log based. And each metadata change is written to the file “edits_inprogress_xxx”.

2) DataNode:

a) Most frequently used system calls:

```
Strace with -c option result:
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 48.13    0.196000       32667         6           poll
 48.13    0.196000        1960       100           accept
  2.76    0.011225          12       904           write
  0.98    0.004000          10       414           mmap
  0.00    0.000000           0       860       401 read
  0.00    0.000000           0       304           open
  0.00    0.000000         0       404          close
  0.00    0.000000         0       309      107 stat
  0.00    0.000000         0       304          fstat
  0.00    0.000000         0        42         lseek
  0.00    0.000000         0      2945           mprotect
  0.00    0.000000         0       809          rt_sigprocmask
  0.00    0.000000         0       551          sched_yield
  0.00    0.000000         0       200          madvise
  0.00    0.000000         0       100          dup2
  0.00    0.000000         0         1        socket
  0.00    0.000000         0         1      1 connect
  0.00    0.000000         0       202          getsockname
  0.00    0.000000         0       102          setsockopt
  0.00    0.000000         0         1        getsockopt
  0.00    0.000000         0       203          clone
  0.00    0.000000         0       306          fcntl
  0.00    0.000000         0       200          rename
  0.00    0.000000         0         2        unlink
  0.00    0.000000         0       203          statfs
  0.00    0.000000         0       203          gettid
  0.00    0.000000         0       406          sched_getaffinity
  0.00    0.000000         0       804          epoll_ctl
  0.00    0.000000         0       203          set_robust_list
------ ----------- ----------- --------- --------- ----------------
100.00    0.407225               11089       509 total
```

According to the above statistic, The most frequently used system calls related to file system are 904 write(), 860 read(), 304 open(), 42 lseek(), 200 rename().

b) File system access pattern:

```
20732 22:48:57
stat("/tmp/hadoop-holmesin/dfs/data/current/BP-1056811197-127.0.1.1-1490748804937/current/rb
w/blk_1073742028", 0x7ff10ebd3200) = -1 ENOENT (No such file or directory)
20732 22:48:57
open("/tmp/hadoop-holmesin/dfs/data/current/BP-1056811197-127.0.1.1-1490748804937/current/r
bw/blk_1073742028", O_RDWR|O_CREAT|O_EXCL, 0666) = 240
20732 22:48:57 fstat(240, {st_mode=S_IFREG|0664, st_size=0, ...}) = 0
20732 22:48:57 close(240)              = 0
20732 22:48:57
open("/tmp/hadoop-holmesin/dfs/data/current/BP-1056811197-127.0.1.1-1490748804937/current/r
bw/blk_1073742028_1204.meta", O_RDWR|O_CREAT, 0666) = 240
20732 22:48:57 fstat(240, {st_mode=S_IFREG|0664, st_size=0, ...}) = 0
20732 22:48:57
open("/tmp/hadoop-holmesin/dfs/data/current/BP-1056811197-127.0.1.1-1490748804937/current/r
bw/blk_1073742028", O_RDWR|O_CREAT, 0666) = 241
……
20732 22:48:57 read(239,
"\0\0\0\24\0\31\t\0\0\0\0\0\0\0\0\21\0\0\0\0\0\0\0\0\30\0%\f\0\0\0\360\377r\222hello world\n", 512) =
47
20732 22:48:57 write(241, "hello world\n", 12) = 12
……
20733 22:48:57
rename("/tmp/hadoop-holmesin/dfs/data/current/BP-1056811197-127.0.1.1-1490748804937/curren
t/rbw/blk_1073742028_1204.meta",
"/tmp/hadoop-holmesin/dfs/data/current/BP-1056811197-127.0.1.1-1490748804937/current/finalize
d/subdir0/subdir0/blk_1073742028_1204.meta") = 0
20733 22:48:57
rename("/tmp/hadoop-holmesin/dfs/data/current/BP-1056811197-127.0.1.1-1490748804937/curren
t/rbw/blk_1073742028",
"/tmp/hadoop-holmesin/dfs/data/current/BP-1056811197-127.0.1.1-1490748804937/current/finalize
d/subdir0/subdir0/blk_1073742028") = 0
```

By observing the strace result on datanode, it will first create two file blk_xxx and the metadata file for it blk_xxx.meta under the folder .../rbw(Replica Being Written). Then it will read the "hello world" from socket, then write this string to the block file "blk_xxx". After the write operation completed, the datanode will rename the two file from folder .../rbw to the finalized folder ".../finalized/...".

**3.6 File system access pattern when creating folders:**

```
#!/bin/bash
for num in {1..2}
do
        hdfs dfs -mkdir /test$num
done
```

The test try to create 100 new folders with the name like /testxx

1) NameNode:
    a) Most frequently used system calls:

    from strace with -c result, it has 1402 read(), 154 lseek(), 517 write(), and 100 fdatasync().

    b) Access pattern: for one mkdir operation, the metadata is written to "edits_inprogress_xxx", and flushed to disk.
2) DataNode:
    a) Most frequently used system calls: from strace with -c result, there are 179 newfstatat(), 12 open(), 7 read(), 3 write().
    b) Access pattern: When creating folders doesn't have much file system calls.

**File system access pattern when change metadata:**

```
#!/bin/bash
for num in {1..100}
do
        hdfs dfs chmod 777 /test/$num.txt
done
```

The test try to change 100 files with a different access permission

3) NameNode:
    a) Most frequently used system calls:

    from strace with -c result, it has 1145 read(), 415 write(), 122 lseek(),and 100 fdatasync().

    b) Access pattern: for one chmod operation, namenode write() the change to the socket, then the new metadata is appened to "edits_inprogress_xxx", and flushed to disk. Notice the lseek() before fdatasync() try to locate the file handle to the current position, each metadata change in appended to the end of the "edits_inprogress_xxx" file, which confirms that the metadata is not overwritten, and it is log based.
4) DataNode: Changing of file access permission doesn't involve datanode operations in file system calls.

**3.7 Important File system access pattern found during testing(checkpointing):**

Access pattern:

```
4607  22:51:01 read(220,
"\33\10\2\20\0\30\320\1\"\0202\321\366z\333\6Mx\265Xt(H\373C\324(\0H\n\vrollEditLog\0227org.a
pache.hadoop.hdfs.server.protocol.NamenodeProtocol\30"..., 102) = 102
……
4620  22:51:01 write(191, "2017-03-28 22:51:01,665 INFO
org.apache.hadoop.hdfs.server.namenode.FSNamesystem: Roll Edit Log from"..., 111) = 111
4620  22:51:01 write(191, "2017-03-28 22:51:01,665 INFO
org.apache.hadoop.hdfs.server.namenode.FSEditLog: Rolling edit logs\n", 97) = 97
4620  22:51:01 write(191, "2017-03-28 22:51:01,665 INFO
org.apache.hadoop.hdfs.server.namenode.FSEditLog: Ending log segment 6\n", 100) = 100
4620  22:51:01 lseek(208, 0, SEEK_CUR)  = 123398
4620  22:51:01 fstat(208, {st_mode=S_IFREG|0664, st_size=1048576, ...}) = 0
4620  22:51:01 write(208, "\27\0\0\0\f\0\0\0\0\0\0\5\333LB\352\r", 17) = 17
4620  22:51:01 fdatasync(208)          = 0
4620  22:51:01 write(191, "2017-03-28 22:51:01,666 INFO
org.apache.hadoop.hdfs.server.namenode.FSEditLog: Number of transaction"..., 231) = 231
4620  22:51:01 lseek(208, 0, SEEK_CUR)  = 123415
4620  22:51:01 fstat(208, {st_mode=S_IFREG|0664, st_size=1048576, ...}) = 0
4620  22:51:01 lseek(208, 0, SEEK_CUR)  = 123415
4620  22:51:01 ftruncate(208, 123415)   = 0
4620  22:51:01 lseek(208, 123415, SEEK_SET) = 123415
4620  22:51:01 close(208)              = 0

……

4620  22:51:01
rename("/tmp/hadoop-holmesin/dfs/name/current/edits_inprogress_0000000000000000006",
"/tmp/hadoop-holmesin/dfs/name/current/edits_0000000000000000006-0000000000000001499")
= 0
4620  22:51:01 write(191, "2017-03-28 22:51:01,667 INFO
org.apache.hadoop.hdfs.server.namenode.FSEditLog: Starting log segment "..., 108) = 108
4620  22:51:01
open("/tmp/hadoop-holmesin/dfs/name/current/edits_inprogress_0000000000000001500",
O_RDWR|O_CREAT, 0666) = 208

……

4620  22:51:01 open("/tmp/hadoop-holmesin/dfs/name/current/seen_txid.tmp",
O_WRONLY|O_CREAT|O_TRUNC, 0666) = 221
……
4620  22:51:01 rename("/tmp/hadoop-holmesin/dfs/name/current/seen_txid.tmp",
"/tmp/hadoop-holmesin/dfs/name/current/seen_txid") = 0
……

4604  22:51:02
open("/tmp/hadoop-holmesin/dfs/name/current/fsimage.ckpt_0000000000000001499",
O_WRONLY|O_CREAT|O_TRUNC, 0666) = 222
4604  22:51:02 fstat(222, {st_mode=S_IFREG|0664, st_size=0, ...}) = 0
4604  22:51:02 write(222, "HDFSIMG1\26\10\213\305\322D\20\350\7\30\337\t
\0(\367\201\200\200\0040\333\v\6\10\376\201\1\20/\10\2\20\201\200\1\32\0*%\10\262\244\334\27
7\261+\20\377\377\377\377\377\377\377\377\177\30\377\377\377\377\377\377\377\377\377\1!\355
\1\2\0\0\1\0\0004\10\2\20\321\201\1\32\4test*"..., 3358) = 3358
4604  22:51:02 fsync(222)              = 0
```

```
4604  22:51:02 close(222)             = 0
……
4604  22:51:02
open("/tmp/hadoop-holmesin/dfs/name/current/fsimage_0000000000000001499.md5.tmp",
O_WRONLY|O_CREAT|O_TRUNC, 0666) = 222
……
4604  22:51:02
rename("/tmp/hadoop-holmesin/dfs/name/current/fsimage_0000000000000001499.md5.tmp",
"/tmp/hadoop-holmesin/dfs/name/current/fsimage_0000000000000001499.md5") = 0
4604  22:51:02
rename("/tmp/hadoop-holmesin/dfs/name/current/fsimage.ckpt_0000000000000001499",
"/tmp/hadoop-holmesin/dfs/name/current/fsimage_0000000000000001499") = 0

……

4604  22:51:02 unlink("/tmp/hadoop-holmesin/dfs/name/current/fsimage_0000000000000000000")
= 0
4604  22:51:02
unlink("/tmp/hadoop-holmesin/dfs/name/current/fsimage_0000000000000000000.md5") = 0
```

When the rolling of edit log happens, the old “edits_inprogress_0000000000000000006” is renamed to a edits log file with start and end number “edits_0000000000000000006-0000000000000001499”. And a new “edits_inprogress_xxx” file is created. New “fsimage.ckpt_xxx”(ckpt stands for checkpoint) and “fsimage_xxx.md5.tmp” are created. Then the namenode try to read from the old edits log file, and then write those log to a new “fsimage_xxx” file. After merging the old fsimage and edits log file, The old fsimage file is then deleted. Fsimage file and md5 file will be renamed.

Checkpointing is the process of merging the most recent fsimage with all edits applied after that fsimage is merged in order to create a new fsimage. After looking into the file hdfs-default.xml

```
<property>
  <name>dfs.namenode.checkpoint.period</name>
  <value>3600</value>
  <description>The number of seconds between two periodic checkpoints.
  </description>
</property>

<property>
  <name>dfs.namenode.checkpoint.txns</name>
  <value>1000000</value>
  <description>The Secondary NameNode or CheckpointNode will create a checkpoint
  of the namespace every 'dfs.namenode.checkpoint.txns' transactions, regardless
  of whether 'dfs.namenode.checkpoint.period' has expired.
  </description>
</property>
```

The checkpointing process is triggered if time has elapsed 3600 since last checkpoint, or if current transactions in edits file has exceeded 1000000.

### 3.8. Possible optimization:

By observation, the namenode will store all the metadata transaction to the file onto the disk. Even for creating a small file, it invokes four fdatasync() I/O. These disk I/O can take a lot of time, which affects the response time to client. One possible optimization would be, batch some writes within a single disk I/O, since these metadata log are continuous in the file.