Using HDFS as a WLCG SE

Brian Bockelman

Introducing Hadoop

- Hadoop is a data processing system that follows the MapReduce paradigm for scalable data analysis.

  - Largest install is at Yahoo, a major contributor.

  - 14PB of online disk.

  - Larger clusters are planned

Hadoop and HDFS

- To do large-scale data processing, you need an underlying file system.

- But to do this affordably, you need a distributed FS designed for commodity hardware.

  - I.e., stuff all your worker nodes full of disks.

HDFS

- HDFS is a scalable file system with two major components:

  - Namenode: central metadata server.

  - Datanode: file servers for data.

- Lots of design decisions in HDFS will look familiar to WLCG sites.

HDFS Design

- Big subject! See the Hadoop whitepapers

- The filesystem keeps all namespace information persisted in a journal and merges the journal once every hr or 64MB.

  - All operations that do not alter namespace are guaranteed to be RAM-only.

  - Benchmarked at 50k ops / sec for reads, 5k ops / sec for writes.
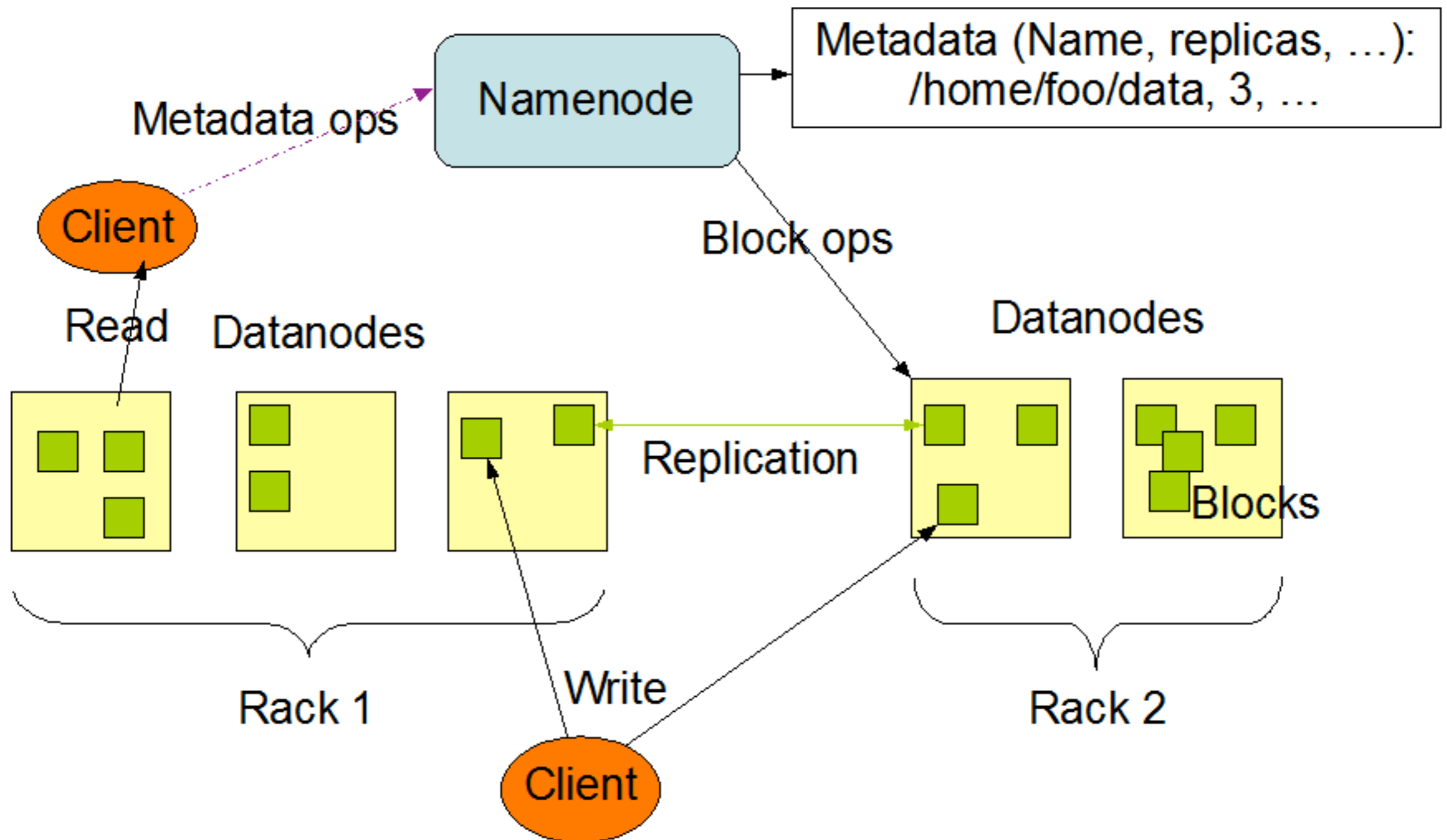
HDFS Architecture

Image courtesy of Hadoop website

HDFS Replication

- Replication built into core of system.

- Default replication policy:

  - First replica to local datanode

  - Second replica to a node on a different rack

  - Third replica to yet another rack.

Replication Example

- Our current policy is that any node that does not have a heartbeat in 10 minutes is declared dead.

- At that point, namenode will start creating new replicas, assuming the node is dead.

- Example below: 1.5TB HDD failed; "danger zone" passed in ~ 1 hr.
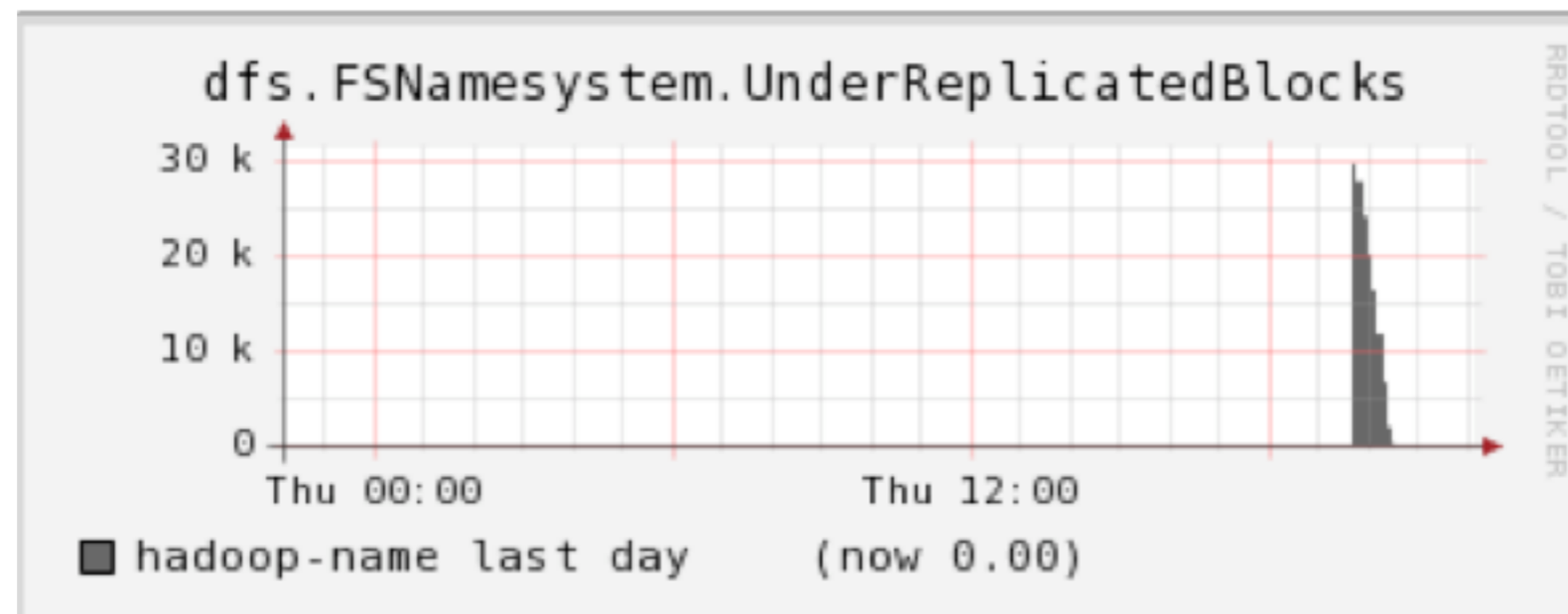
Replication

- At no point when a HDD fails does a client fail!

HDFS Replication

- HDFS replication allows client's reads to survive:

  - Death of datanode currently reading from.

  - Death of namenode.

Grid-Enabling HDFS

- We combine HDFS with two grid components:

  - BestMan SRM server

  - Globus Gridftp

  - Both are well maintained & modular.

- And then we mount it on our WN for local file access.

HDFS SE Diagram

HDFS Grid Site
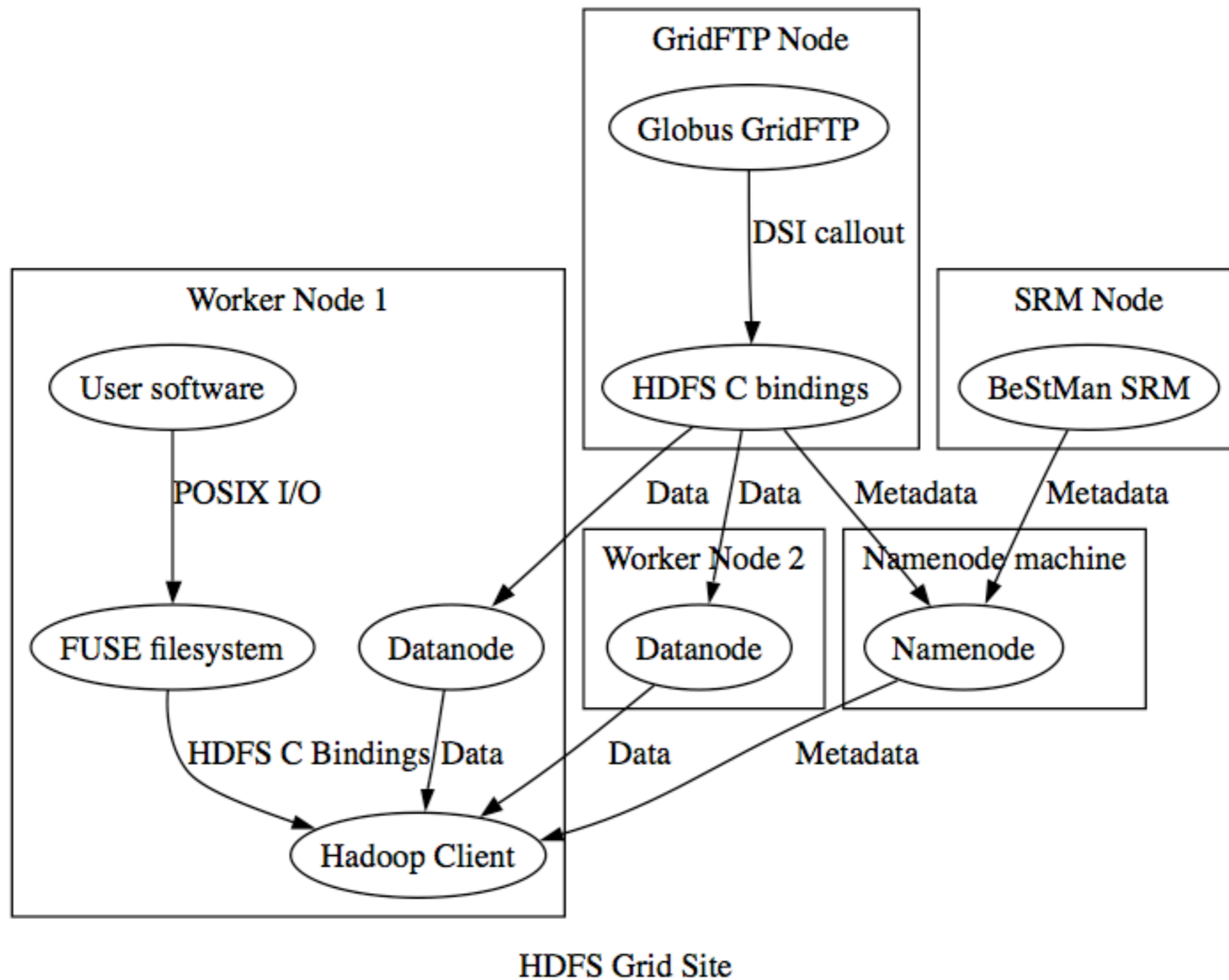
Advantages of HDFS

- In order, these are the primary drivers of our use of HDFS:

  - Manageability

  - Reliability

  - Usability

  - Scalability

Manageability

- The following tasks are trivial:
    - Integration of statistics with **Ganglia**.
    - **Decommissioning** hardware.
    - **Recovery** from hardware failure.
    - **Fsck**!
        - Checks the current knowledge of the filesystem and counts how many block replicas there are per file, and highlights any which are under-replicated.
    - RPM and Pacman-based install for the whole kit.
    - Many of our "well-known" problems are not possible.
        - **Don't need a separate admin toolkit**!
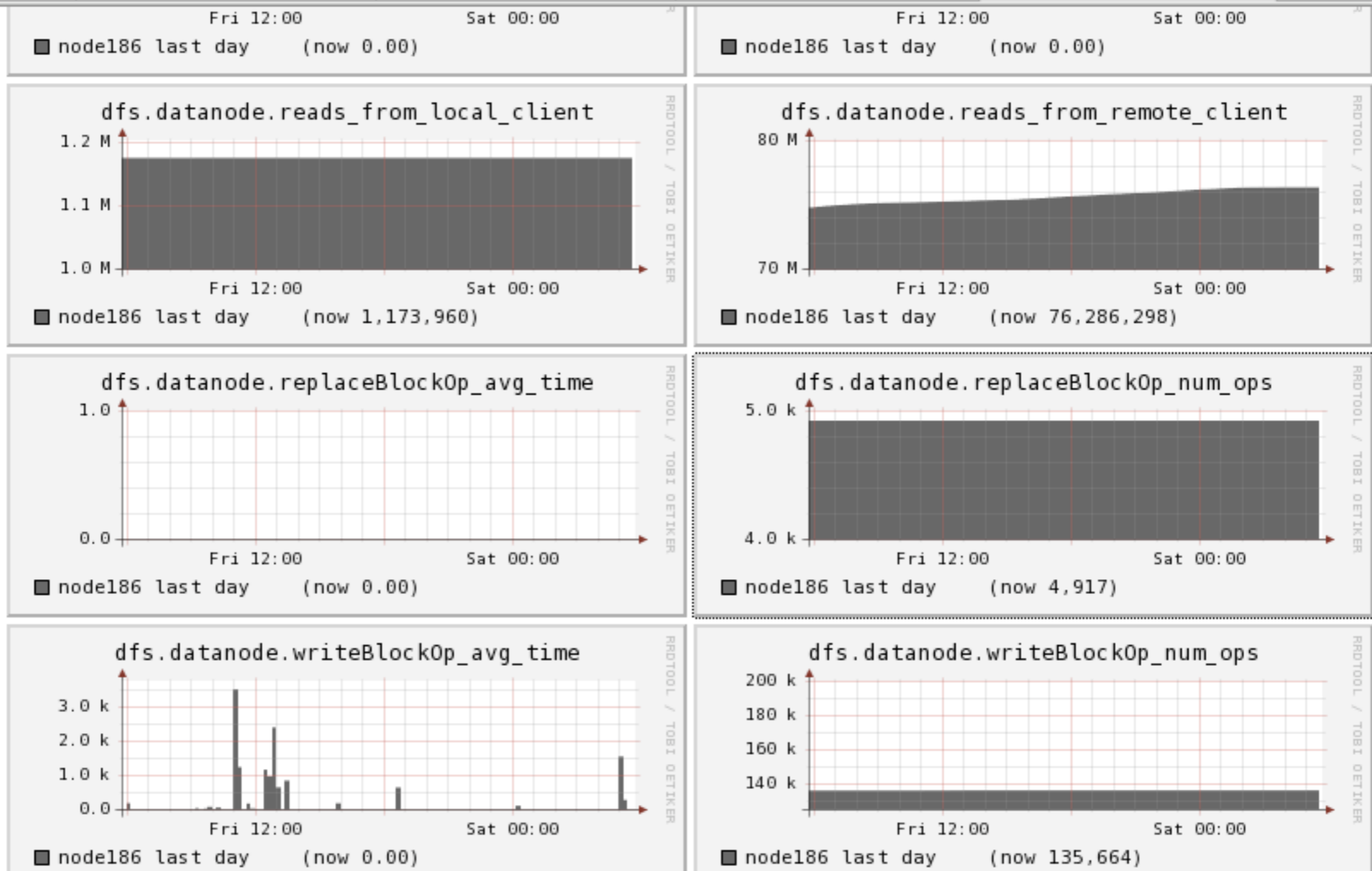   - Setting quotas.

FSCK example


```
root@hadoop-name:~ — ssh — 107×33

.................................................................................
.................................................................................
.................................................................................
.................................................................................
.................................................................................
.................................................................................
.................................................................................
.................................................................................
.................................................................................
.................................................................................
....................................................................Status: HEALTHY
Total size:     72767054047268 B
Total dirs:     2271
Total files:    59765 (Files currently being written: 1)
Total blocks (validated):       1053128 (avg. block size 69096115 B)
Minimally replicated blocks:    1053128 (100.0 %)
Over-replicated blocks:         3778 (0.3587408 %)
Under-replicated blocks:        0 (0.0 %)
Mis-replicated blocks:          0 (0.0 %)
Default replication factor:     3
Average block replication:      2.0923886
Corrupt blocks:                 0
Missing replicas:               0 (0.0 %)
Number of data-nodes:           113
Number of racks:                1


The filesystem under path '/' is HEALTHY

real    0m7.753s
user    0m0.835s
sys     0m0.159s
[root@hadoop-name ~]#
```

Ganglia Graphs

Ganglia:: node186 Host Report

f.unl.edu/ganglia/?r=day&c=red-workers&h=node186

://nmi-s005.cs.wi... | [#HADOOP-4343] Add... | WLCG Collaboration W... | Results of Query: Job ... | Ganglia:: node186 Ho... | Comput

node186 last day   (now 0.00)

node186 last day   (now 0.00)

dfs.datanode.reads_from_local_client

node186 last day   (now 1,173,960)

dfs.datanode.reads_from_remote_client

node186 last day   (now 76,286,298)

dfs.datanode.replaceBlockOp_avg_time

node186 last day   (now 0.00)

dfs.datanode.replaceBlockOp_num_ops

node186 last day   (now 4,917)

dfs.datanode.writeBlockOp_avg_time

node186 last day   (now 0.00)

dfs.datanode.writeBlockOp_num_ops

node186 last day   (now 135,664)

Reliability

- Replication works incredibly well.

- Client (CMSSW) reads live through restarts of any piece of HDFS.

- Writes are pipelined; guaranteed to have N copies on cluster when close() returns.

- Each datanode does a constant

Usability

- POSIX works*; this opens a lot of doors to communities who are put off by recompiling their software.

    - * = writes are append-only

- Users no longer have to know about your FS-specific tools.

    - Although there are some nifty additions the tools provide.

(CMSSW) Performance
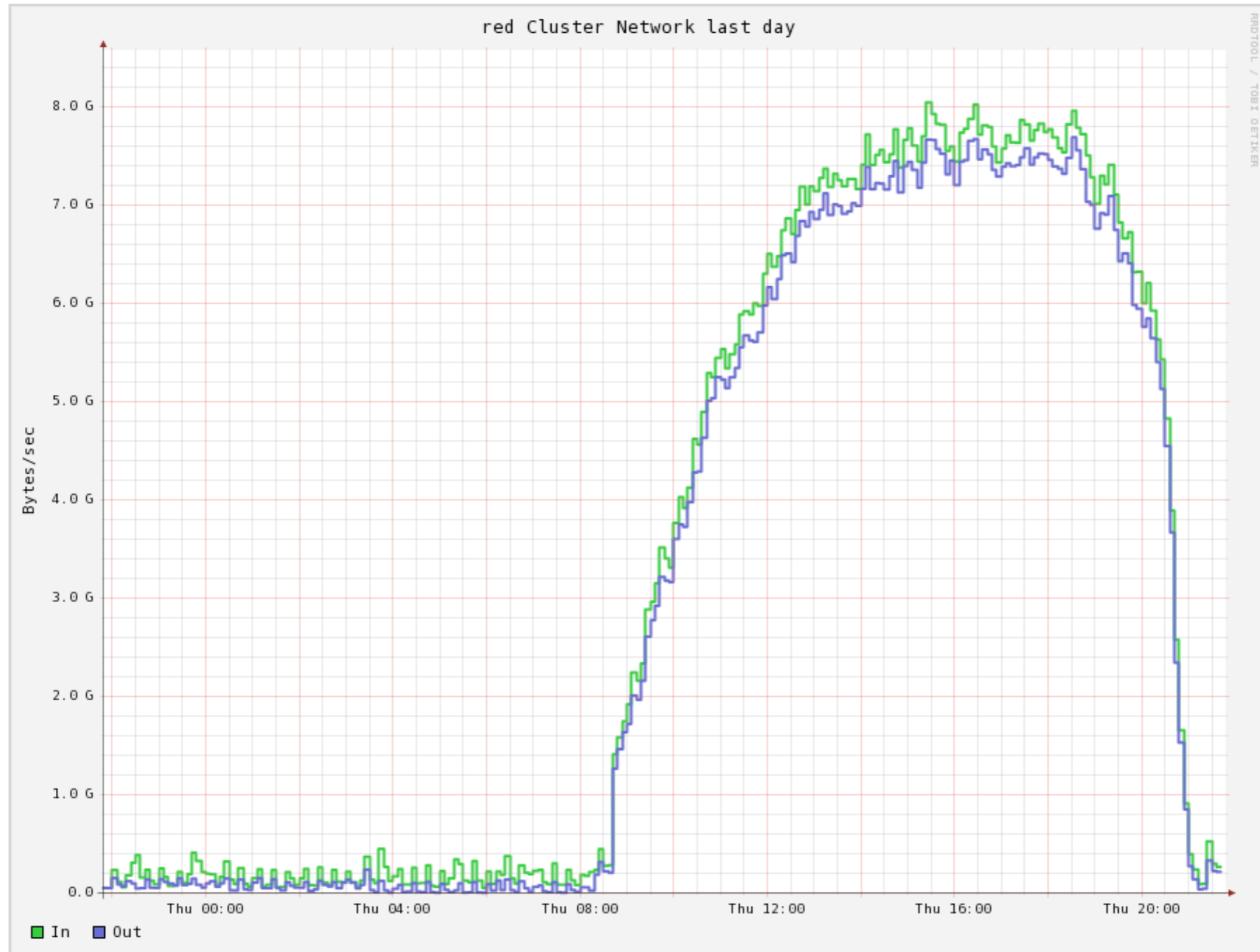
Performance Stats

- We've clocked:

  - The filesystem at 80Gbps

  - 23 Gbps for 300 CMSSW processes analyzing a *single file* @ 2 replicas (we picked a fake workflow to pump up the per-job rate).

  - SRM endpoints at 30Hz (these SRMs are stateless; load-balancing is trivial)

  - fsck takes <10s

  - Decommissioning a pool <1hr.

  - Namenode restart in about 60s

  - WAN transfers peak at 9Gbps, sustain 5Gbps.

Conclusions

- Hadoop gives us significant improvements in manageability of storage => lowers cost of maintenance.

- Performance scalability benefits by co-locating storage and WNs

  - Reliability during disk failures => less failures seen by users.

- Allows us to use commodity hardware => lowers cost of hardware.