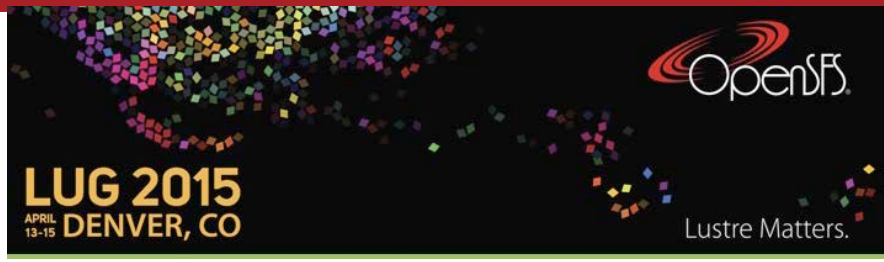
Computer simulations create the future



Analysis and Elimination of Client Evictions on a Large Scale Lustre Based File System

Keiji Yamamoto*, Fumiyoshi Shoji, Atsuya Uno
RIKEN AICS

Shuji Matsui, Kenichiro Sakai,
Fumichika Sueyasu, Shinji Sumimoto
Fujitsu Limited

Apr.14 2015

K computer

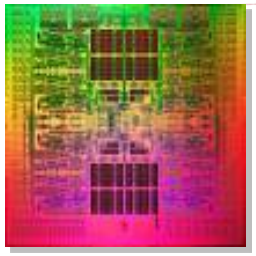
Outline

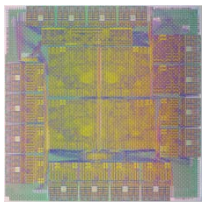
- The K computer and I/O architecture Overview

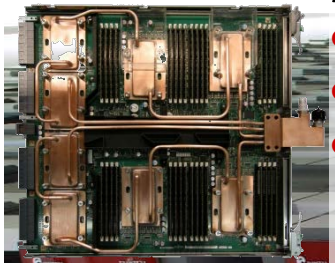
- Eviction problem
  - Analysis on the K computer
  - Reproduction test

- Evaluation on the K computer

System overview of the K computer







Processor: SPARC64™ VIIIfx

- Fujitsu's 45nm technology
- 8 Core, 6MB Cache Memory and MAC on Single Chip
- High Performance and High Reliability with Low Power Consumption

Interconnect Controller (ICC)

- 6 dims-Torus/mesh (Tofu Interconnect)

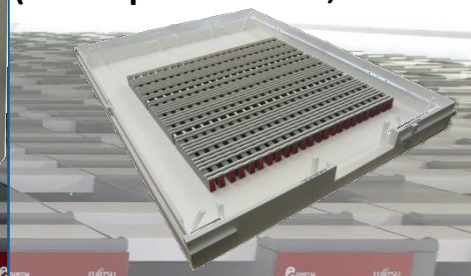
System Board: High Efficient Cooling

- With 4 Computing Nodes
- Water Cooling: Processors, ICCs etc.
- Increasing component lifetime and reducing electric leak current by low temperature water cooling

Rack : High Density

- 96 compute nodes and 6 I/O nodes on Single Rack
  - 24 System Boards
  - 6 IO System Boards
  - System Disk
  - Power Units

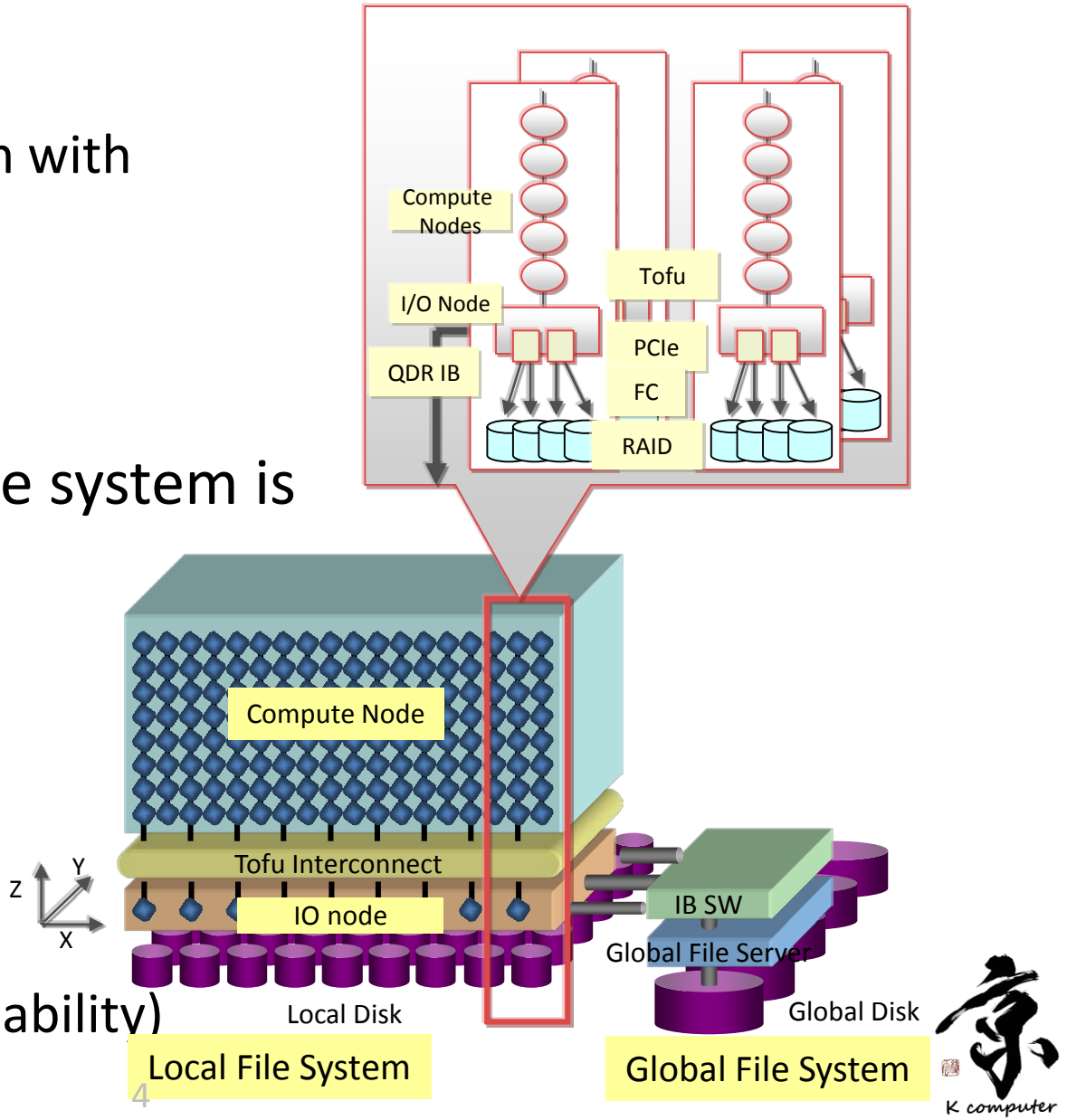
(10PFlops:864 Racks, 82944 nodes)



I/O architecture of the K computer

- File system
  - Based on Lustre File system with several extensions

- Configurations of each file system is optimized for each.
  - Local File System
    2,592-OSSes (5,184 OSTs)
    11PB (for Performance)
  - Global File System:
    90-OSSes (2,880 OSTs)
    30PB (for Capacity and Reliability)



Eviction Problem

- What is the Eviction?
  - File server evicts a client when the client does not works properly, e.g. no response to requests from servers.

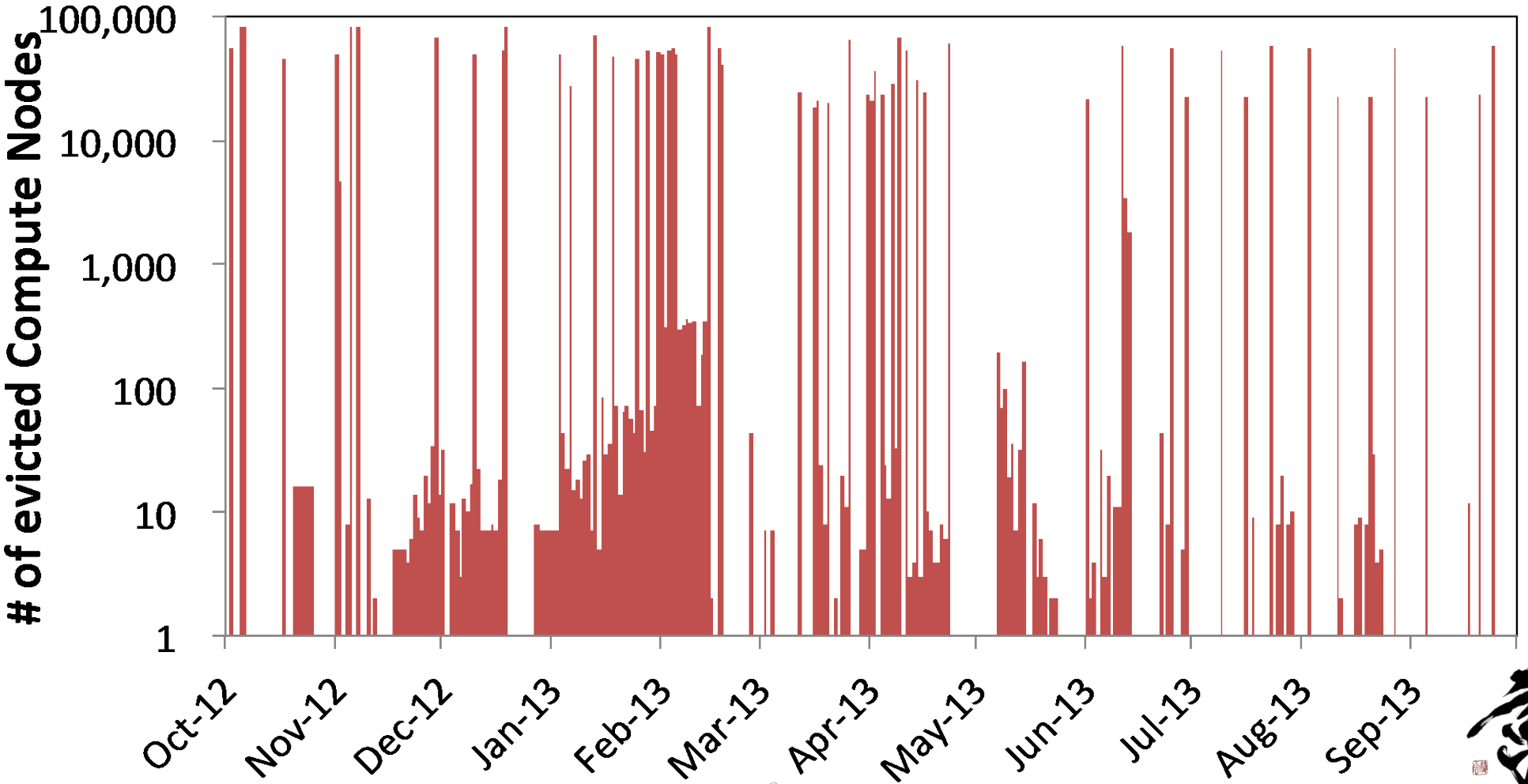
- Influence of the Eviction
  - Since the evictions cause I/O stalls on the client nodes, I/O accesses of running jobs on the nodes will fail.
  - In many cases, the jobs affected by the evictions are aborted.
  - The case of the K computer, many evictions occur frequently and node utilization decreases seriously.

- On the K computer, eviction is one of the most serious issue to be solved.

K computer

Eviction problem on the K computer

- Eviction occurs in several tens of thousand compute node per day

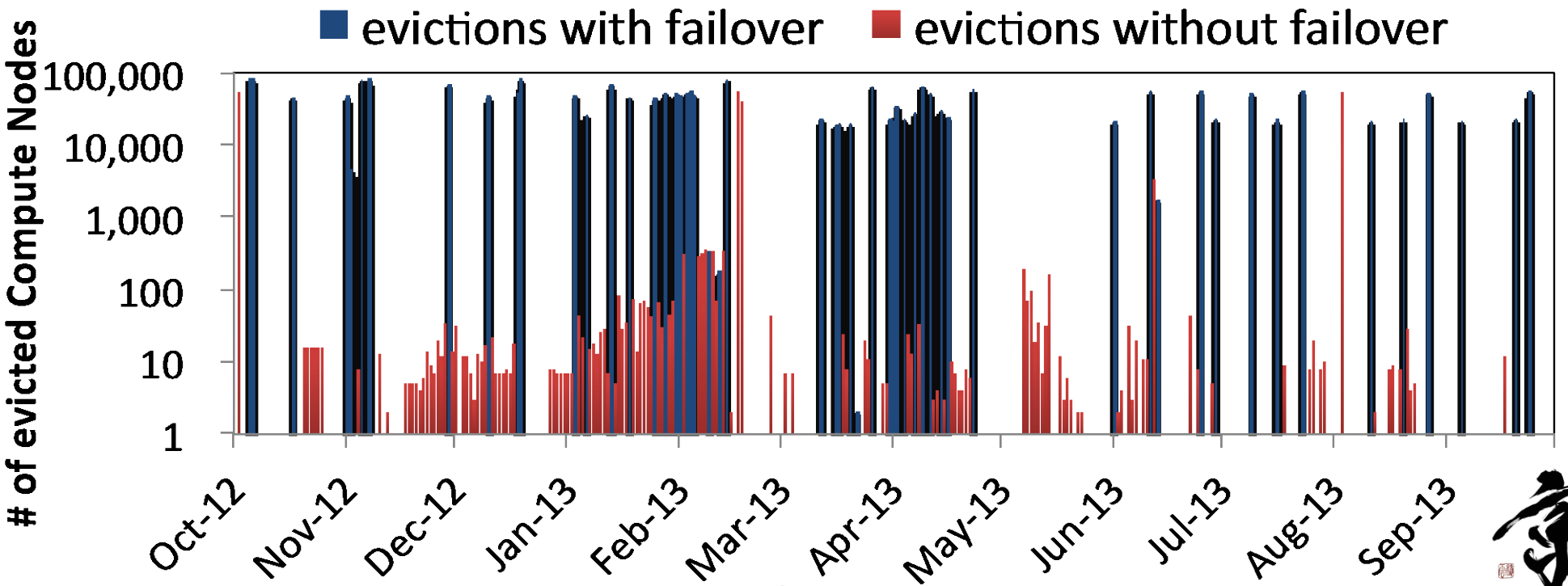


Triggers of Eviction on the K computer

- Many of evictions were related with
  - OSS Failover
    - Hardware failure
      - OSS hardware failure
    - Software failure
      - Software hung-up
      - Timeout
    - Scheduled maintenance
  - System Board Maintenance
    - CPU failure
    - Memory failure
    - Interconnect failure

Eviction and OSS Failover

- We observed strong correlation between large-scale eviction and OSS failover.
  - Longer Failover Time by Network and MDS/OSS Failure
    - Network Link Down
    - LNET Router Down
    - MDS/OSS Down
  - Large-scale eviction (# of evicted Compute nodes > 1,000):
    - 57 times (52 times with failover)



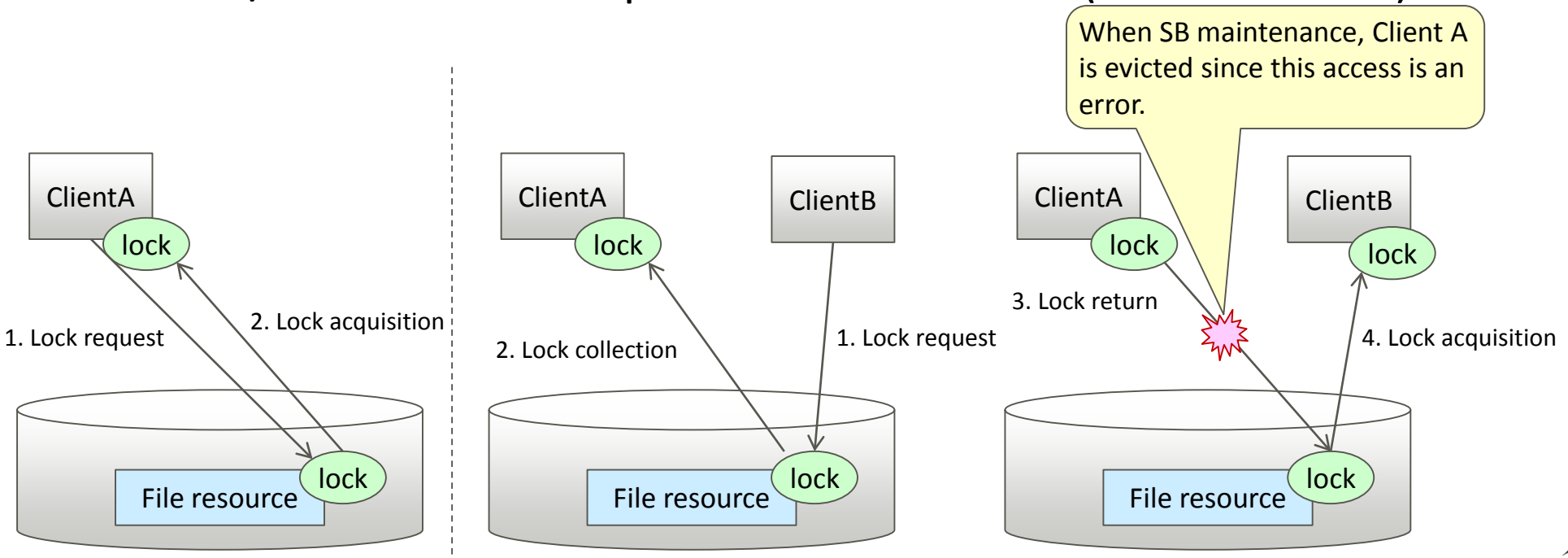
K computer

Triggers of Eviction on the K computer

- Many of evictions were related
  - OSS Failover
    - Hardware failure
      - OSS hardware failure
    - Software failure
      - Software hung-up
      - Timeout
    - Scheduled maintenance
      - Once or Twice a month
  - System Board Maintenance
    - CPU failure
    - Memory failure
    - Interconnect failure

Eviction by Maintenance of System Board(SB)

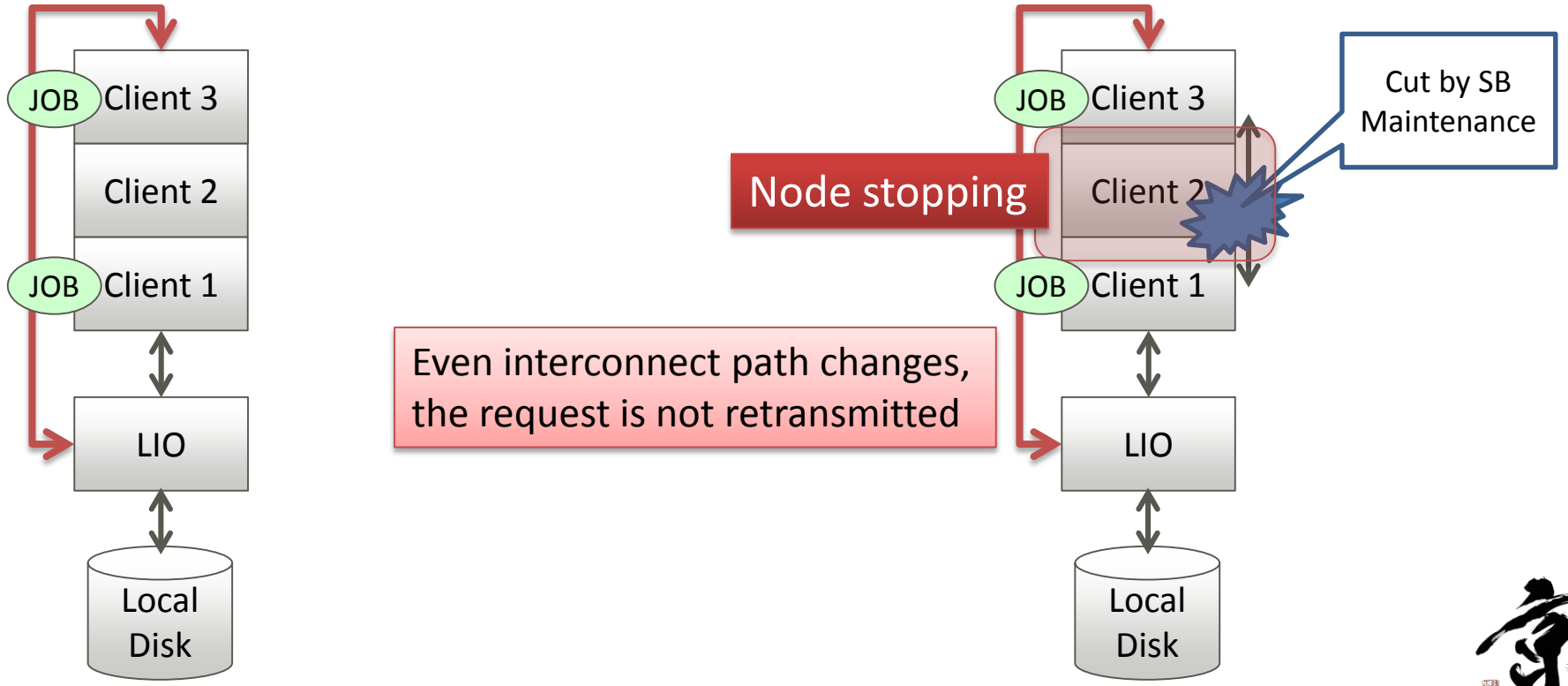
- Eviction occurs by lock acquisition and recovery of between clients in the job
  - When client accesses the same file resources
    - Create a file under the job directory (lock on the parent directory)
    - Read/Write from multiple nodes in same file(lock on extent)



K computer

Interconnect Failure Related to System Board Maintenance

- Interconnect network connection was cut by System Board (SB) maintenance
- Recovery time of Interconnect failover was the issue



K computer

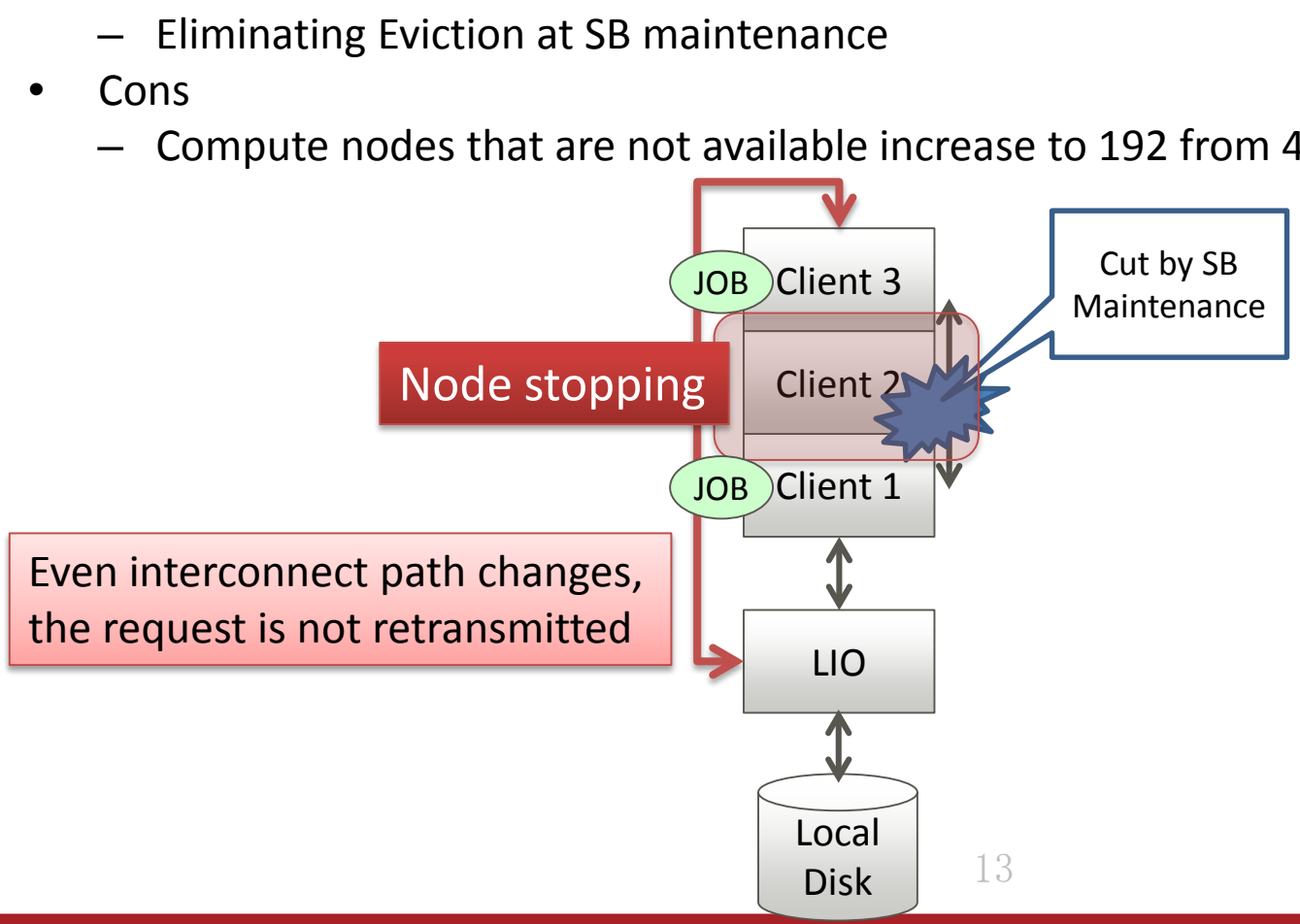
Solutions at SB maintenance

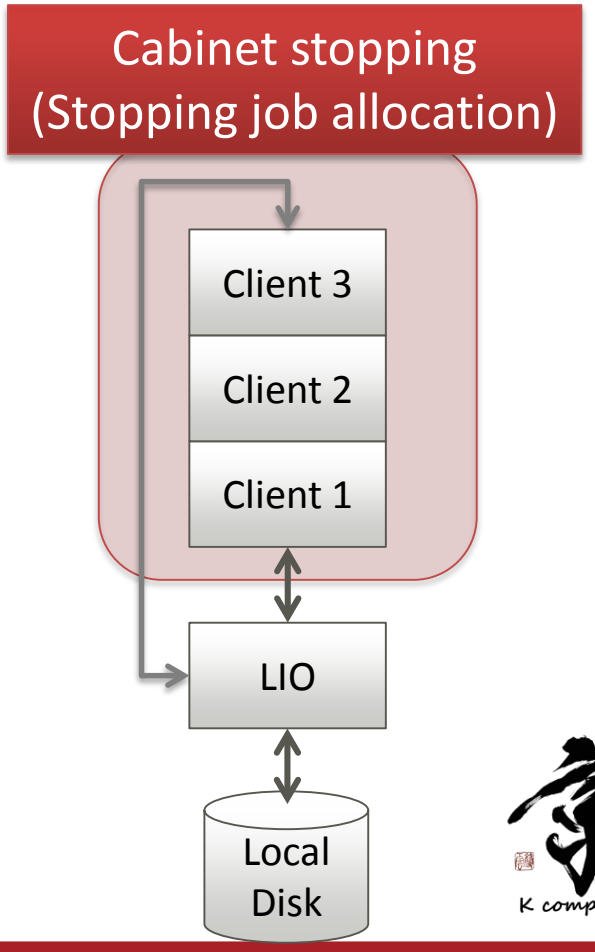
- Step 1: Eliminating Eviction at SB maintenance
  by System Operation Level

- Step 2: Eliminating Eviction at SB maintenance
  by Improvement of File System Level

Step:1 System operation level

- Changing the SB maintenance after each system cabinet stopping
  1. Stopping job allocation to cabinet (192 compute nodes)
  2. Confirming the running jobs to finish
  3. SB maintenance
- Pros
  - Eliminating Eviction at SB maintenance
- Cons
  - Compute nodes that are not available increase to 192 from 4

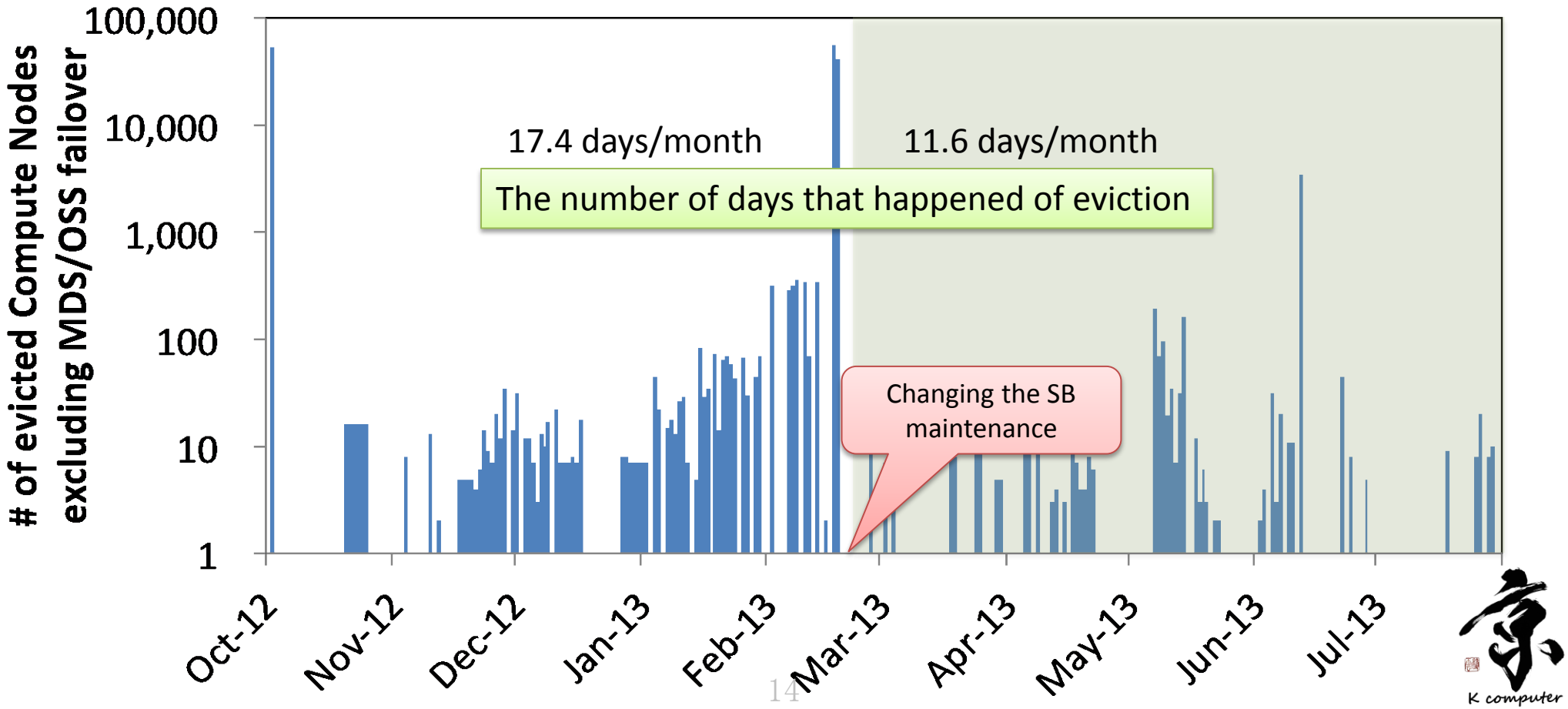




K computer

Evaluation of system operation level

- Eliminating Eviction at SB maintenance by System Operation Level
  - Changing the SB maintenance after each system cabinet stopping



K computer

Step 2: Improvement of File System Level

- Basic Policy against the Timeout:
  - Client Side: Acceleration of RPC Retry
  - Server Side: Keeping Ready to Receive Client RPC

- Under the Basic Policy, we fixed two issues.
  - Issue 1: Clients do not Resend Lock Collection Request under some situation.
    - Adding Sequences to fix the situation and resend
  - Issue 2: Depletion of Server Thread Resource to Receive Client's Reconnect in case of MDS/OSS overload
    - Preparing Dedicated Server Thread Resources

Evaluation of Client Eviction Problem Fix

- Evaluated on our Test Environment (1 Rack)
- Generate the lock contention intentionally
  - Extract tar file including many files in the same path
  - Turn off a System Board
- By the two fixes, eviction occurrence ratio was reduced to 1/72.

	Before	After	Improvement
Ratio	0.47	0.0065	1/72

Eviction Occurrence Ratio/Node

K computer

Summary and Future work

- Eviction is the biggest problem in file system on the K computer
  - OSS Failover
  - SB maintenance

- Analysis and Elimination of Client Evictions
  - Step 1: Eliminating Eviction at SB maintenance by System Operation Level
  - Step 2: Eliminating Eviction at SB maintenance by improvement of File System Level

- Evictions at OSS failover still remain, we are continuing to approach to fix them
  - We have already reconstructed ACT-ACT OSSes during scheduled maintenance for reducing the impact on system


京
K computer

LUG2015

FUJITSU

shaping tomorrow with you

Metadata Access Reduction of Large Scale Lustre Based File System

Apr.14 2015

Shinji Sumimoto*, Shuji Matsui, Kenichiro Sakai, and Fumichika Sueyasu (Fujitsu Limited)
Fumiyoshi Shoji, Atsuya Uno, Keiji Yamamoto (RIKEN AICS)

OpenSFS

Outline

FUJITSU

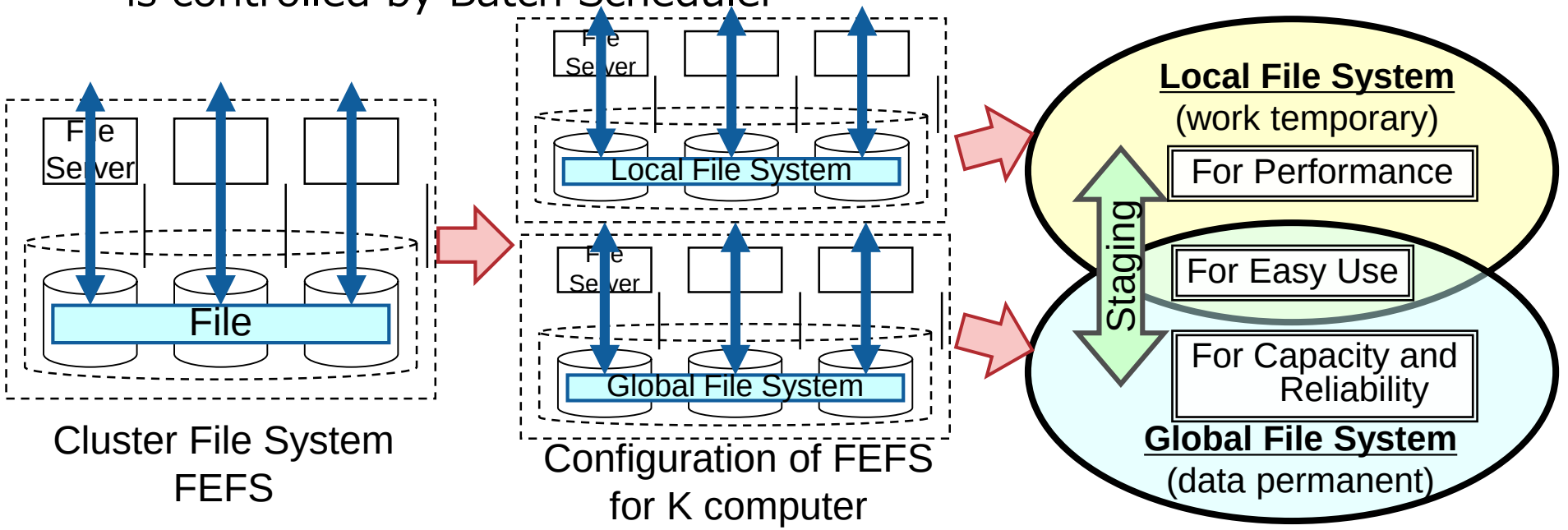
- File System Usage in User Jobs on K computer
- File Access Issues on Local File System
- Meta Data Access Distribution by Loopback File System
- Evaluation on K computer

FUJITSU

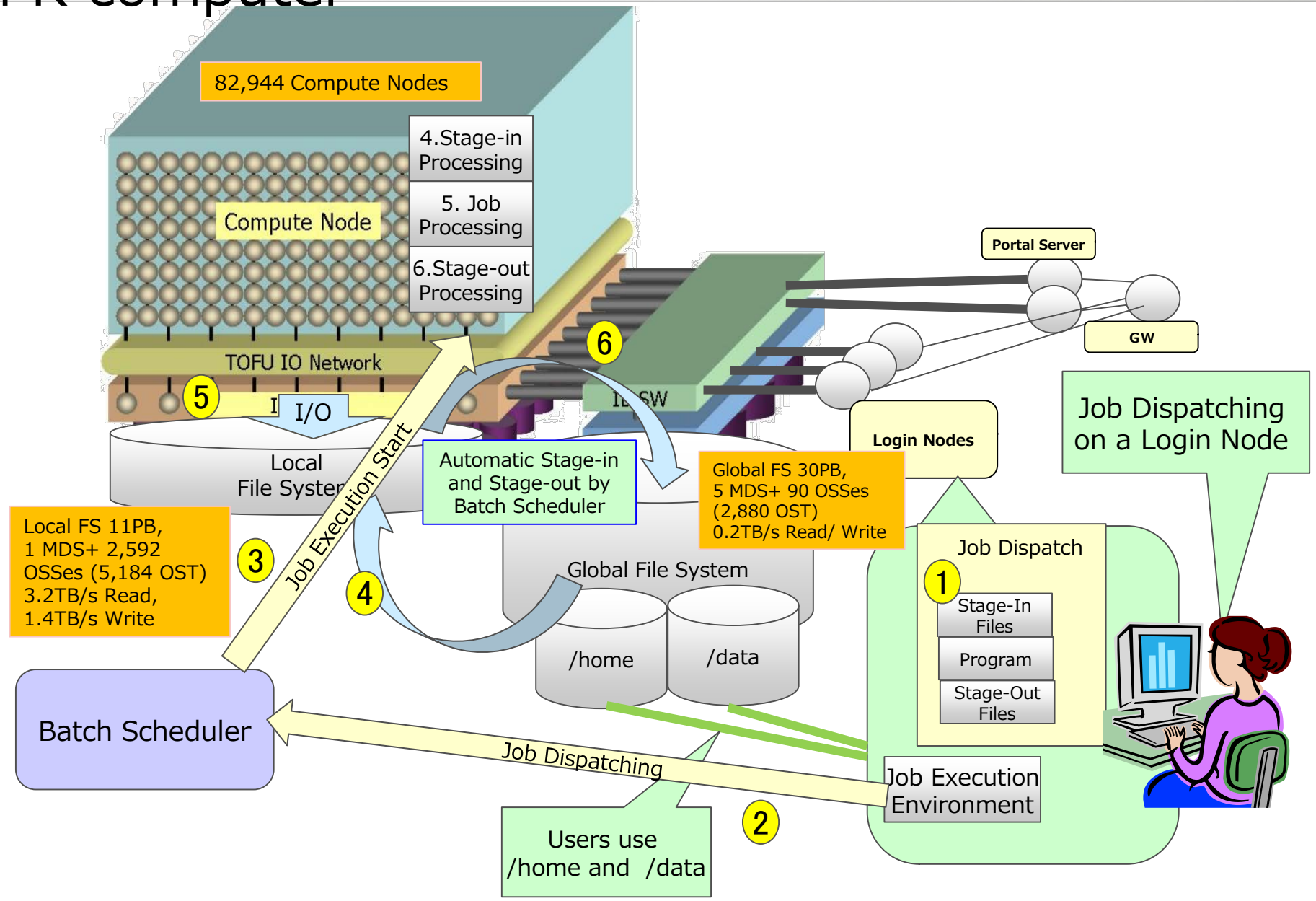
FILE SYSTEM USAGE IN USER JOBS ON K COMPUTER

Overview of FEFS for K computer

- Goals: To realize World Top Class Capacity and Performance File system 100PB, 1TB/s
- Based on Lustre File System with several extensions
  - These extensions are now going to be contributed to Lustre community.
- Introducing Layered File system for each file layer characteristics
  - Temporary Fast Scratch FS(Local) and Permanent Shared FS(Global)
  - Staging Function which transfers between Local FS and Global FS is controlled by Batch Scheduler



Job Execution and File System Accesses on K computer

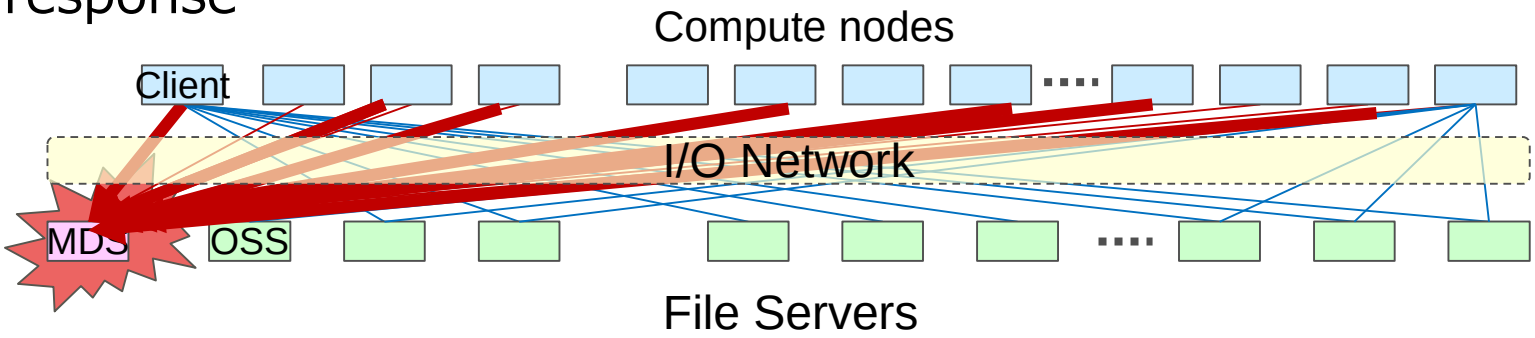


FUJITSU

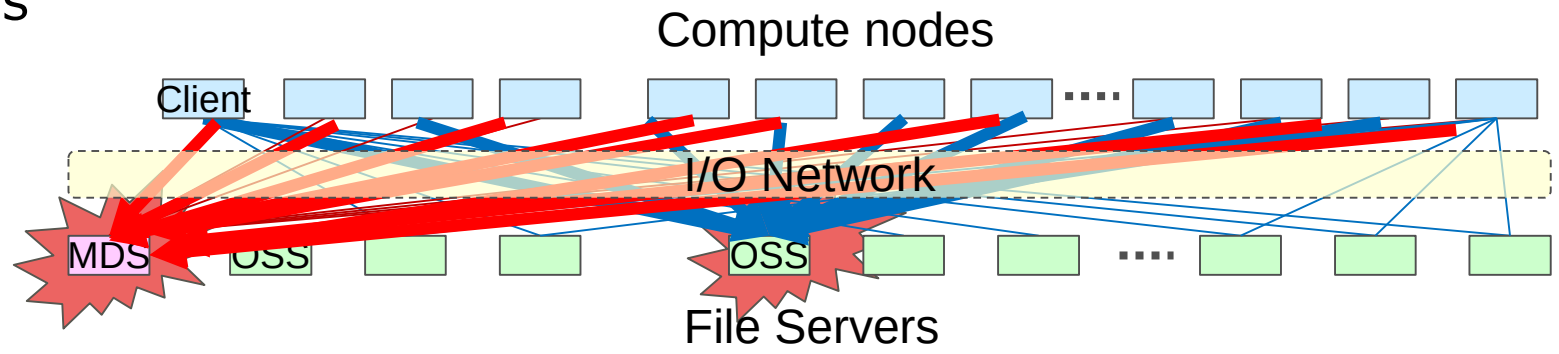
FILE ACCESS ISSUES ON LOCAL FILE SYSTEM

Meta Data Access Issues of Local File System on several 10,000 node job.

- Creating a lot of files per MPI rank at a time.
  - 1,000 file per rank creation becomes 10 M file creation per job.
  - Creating and deleting files take several hours to finish and cause slow MDS response

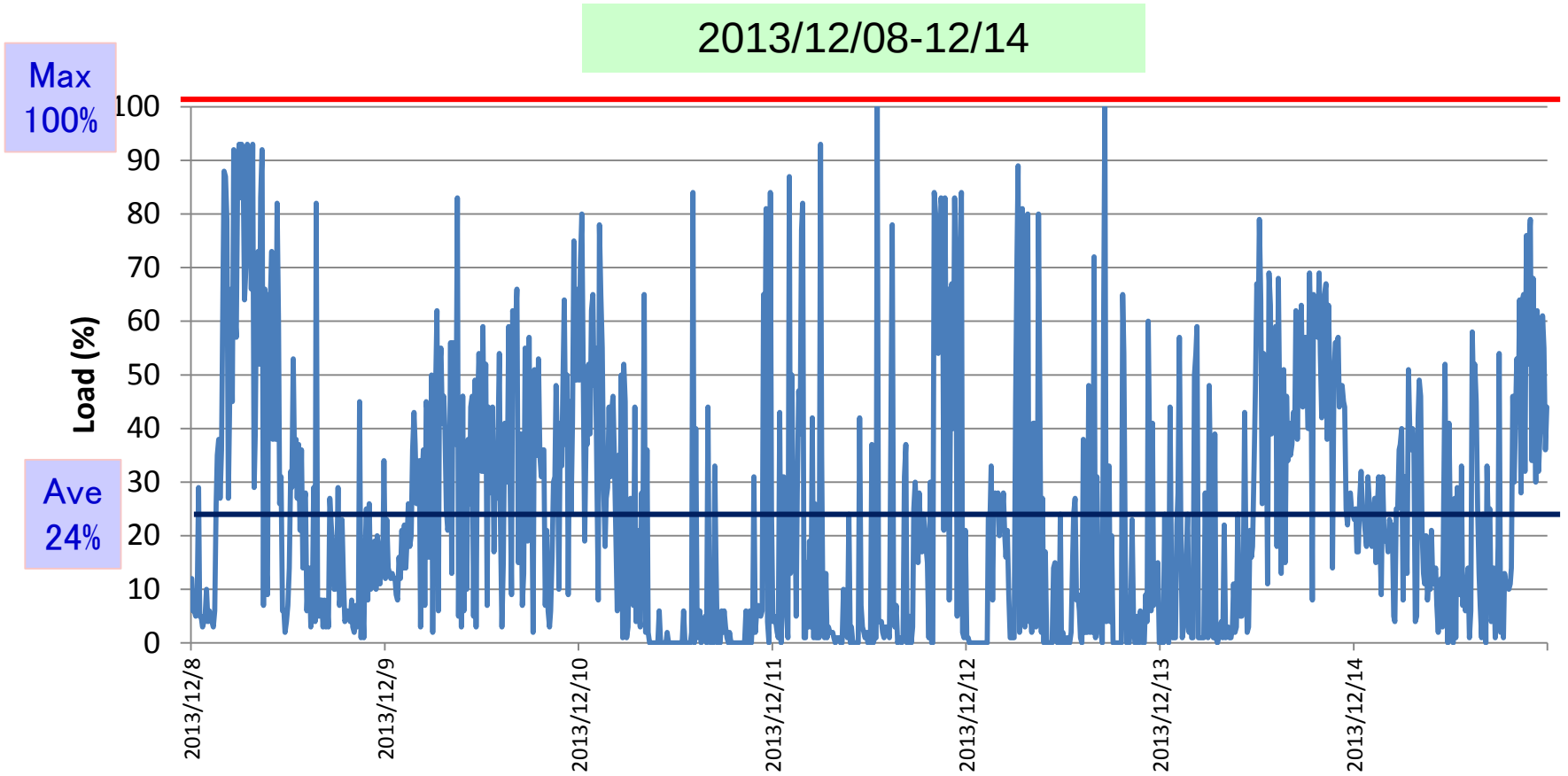


- Execution binaries on shared directory.
  - Concentration access to a single MDS and OST from several 10,000 node takes a long time to finish. Long time delay occurs on starting jobs



MDS CPU Load on Dec. 2013.

- MDS Load was average 24% peak 100% on Dec. 2013.



Issues to Solve and Our Goals

FUJITSU

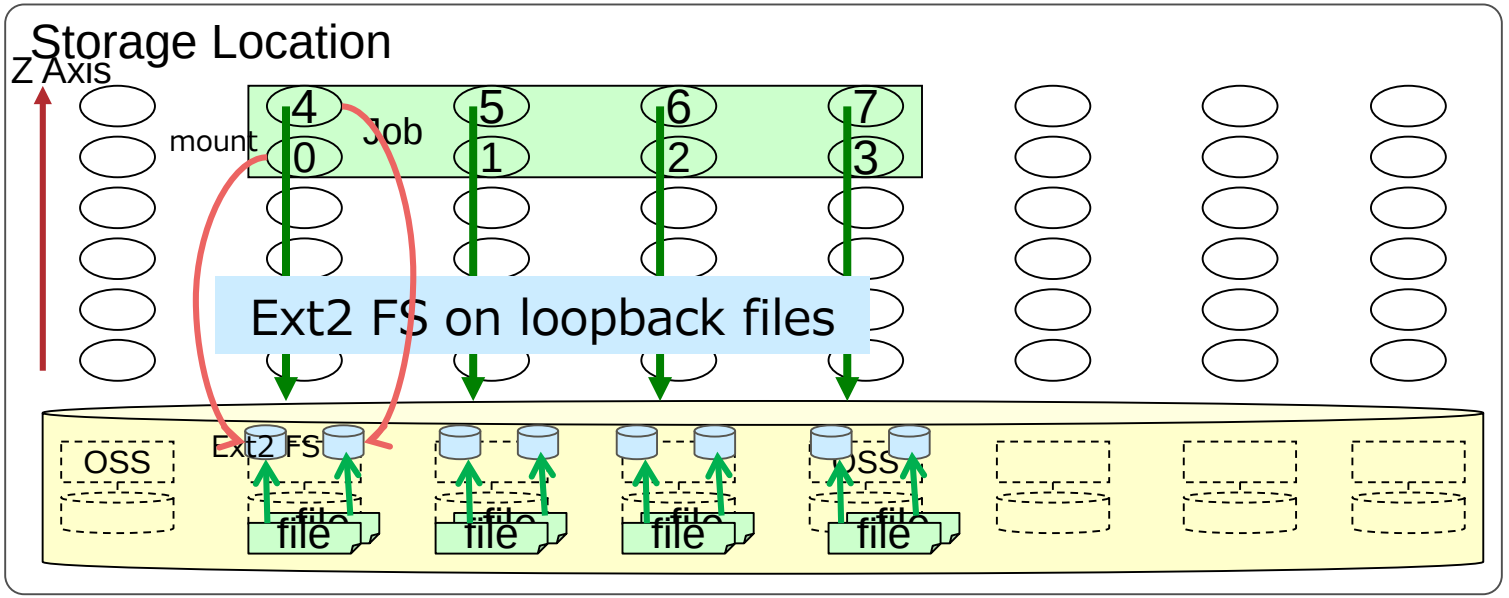
- Issues to Solve:
  - Concentration access to a single MDS or OST on job execution
  - Violent Fluctuations of MDS/OSS load depending on jobs
- Our Goals
  - Distributing and leveraging Meta Data and Data Access
  - Providing faster access performance per MPI rank

FUJITSU

META DATA ACCESS DISTRIBUTION BY LOOPBACK FILE SYSTEM

Meta Data Access Distribution by Loopback File System on K computer

- Providing real local file system per rank by using loopback file
  - Creating loopback file and mounting it as Ext2 file system per MPI rank
  - Rank local data and execution binaries are copied to rank local file system
- Job scheduler software automatically manages creating, mounting and deleting the rank local file system.
  - MDS load can be decreased to only one file creation/deletion per rank
  - No fluctuation and no dependence per Job types (Constant Load)



Comparison of Multiple MDS vs. Loopback

FUJITSU

- We compare the loopback with multiple MDS which could be the other method to solve high load of MDS.

- Multiple MDS(Lustre DNE)
  - Pros:
    - Increasing Meta Data performance on shared file system
  - Cons:
    - Requiring additional hardware resource: MDS, MDT
      Scalability is limited to hardware resource
- Loopback
  - Pros:
    - Completely Scalable Meta Data performance for rank local access
    - No additional hardware
  - Cons:
    - Unable to share among the other nodes
    - Additional Ext2 file system and Loopback Layer Overhead

FUJITSU

EVALUATION ON K COMPUTER

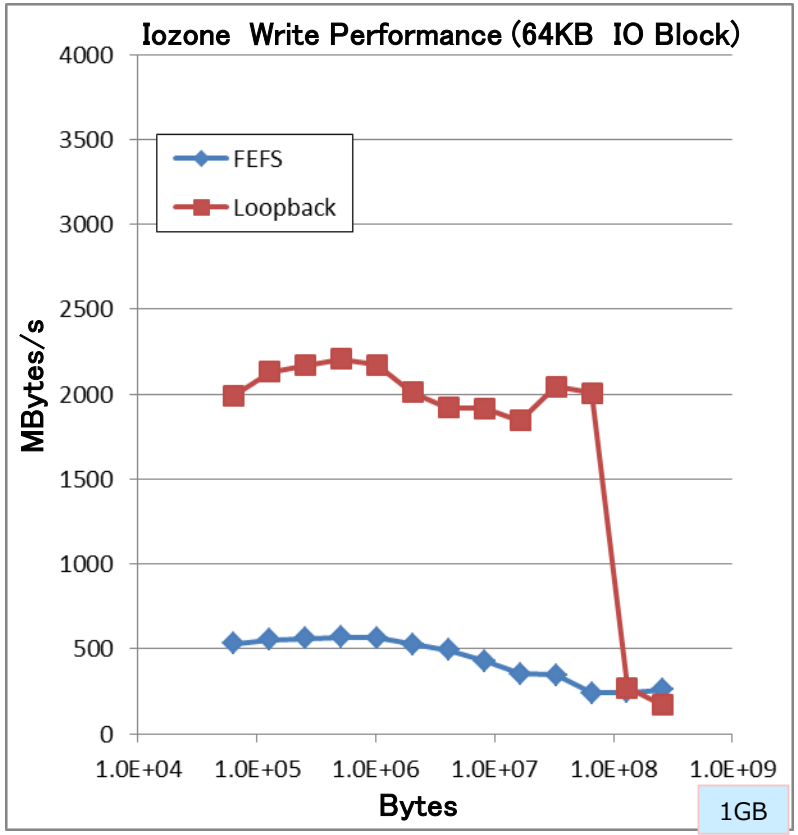
Evaluation of Loopback Based Rank Local File System on K computer

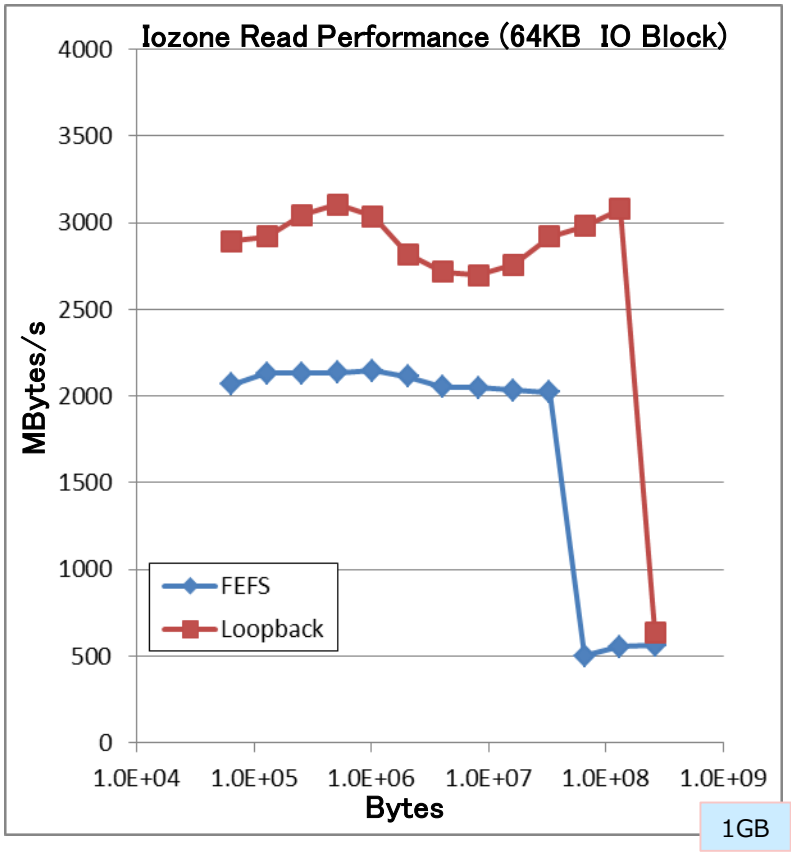
FUJITSU

- Single Node File Access Performance
- Total Meta Data Access Performance
- Comparison of MDS Load (Before vs. After)

Single Node File Access Performance

- Single Node File Write/Read Performance by iozone
- Loopback based file system achieved better performance at small file size by file system cache

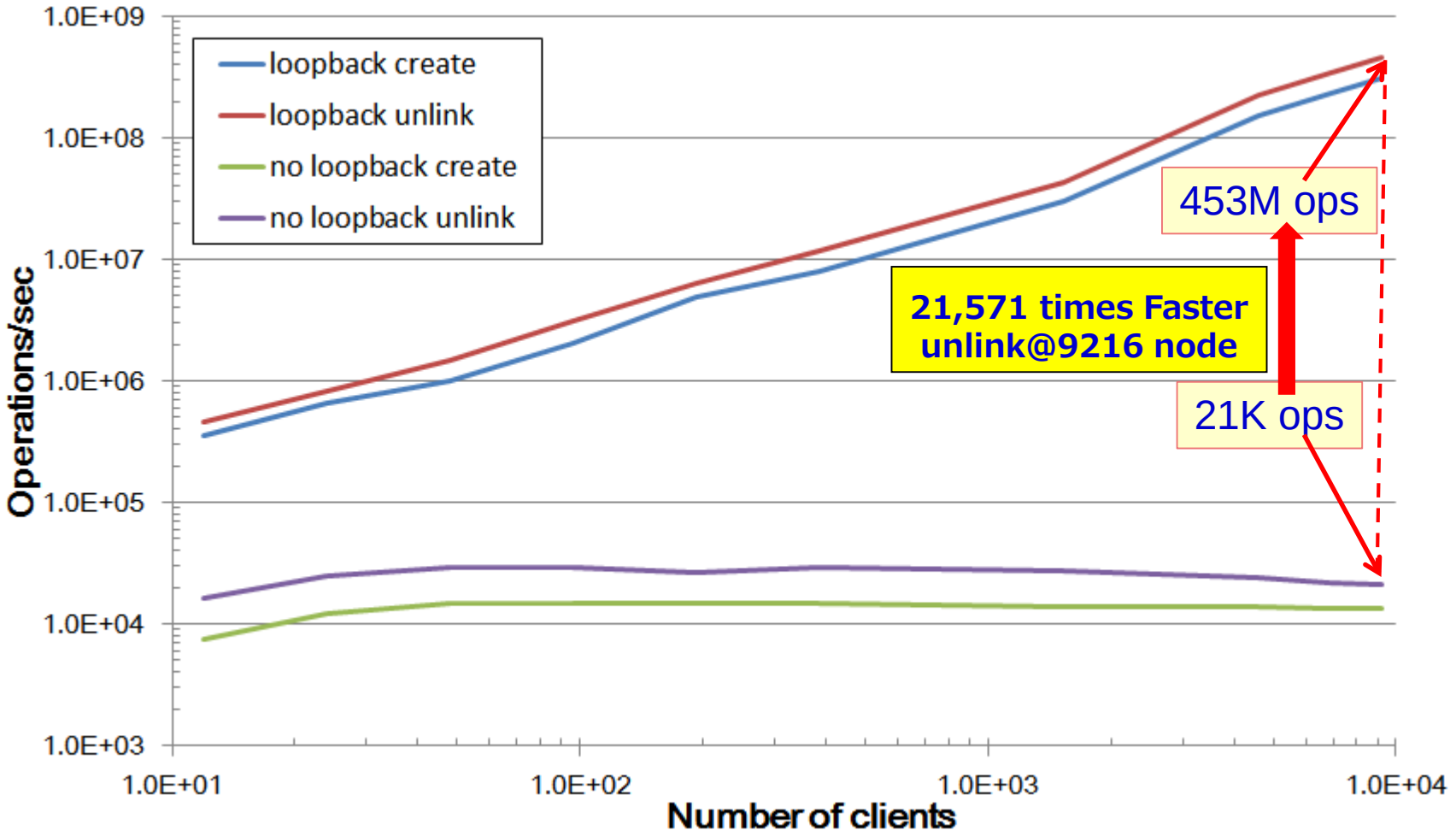




Total Meta Data Access Performance

FUJITSU

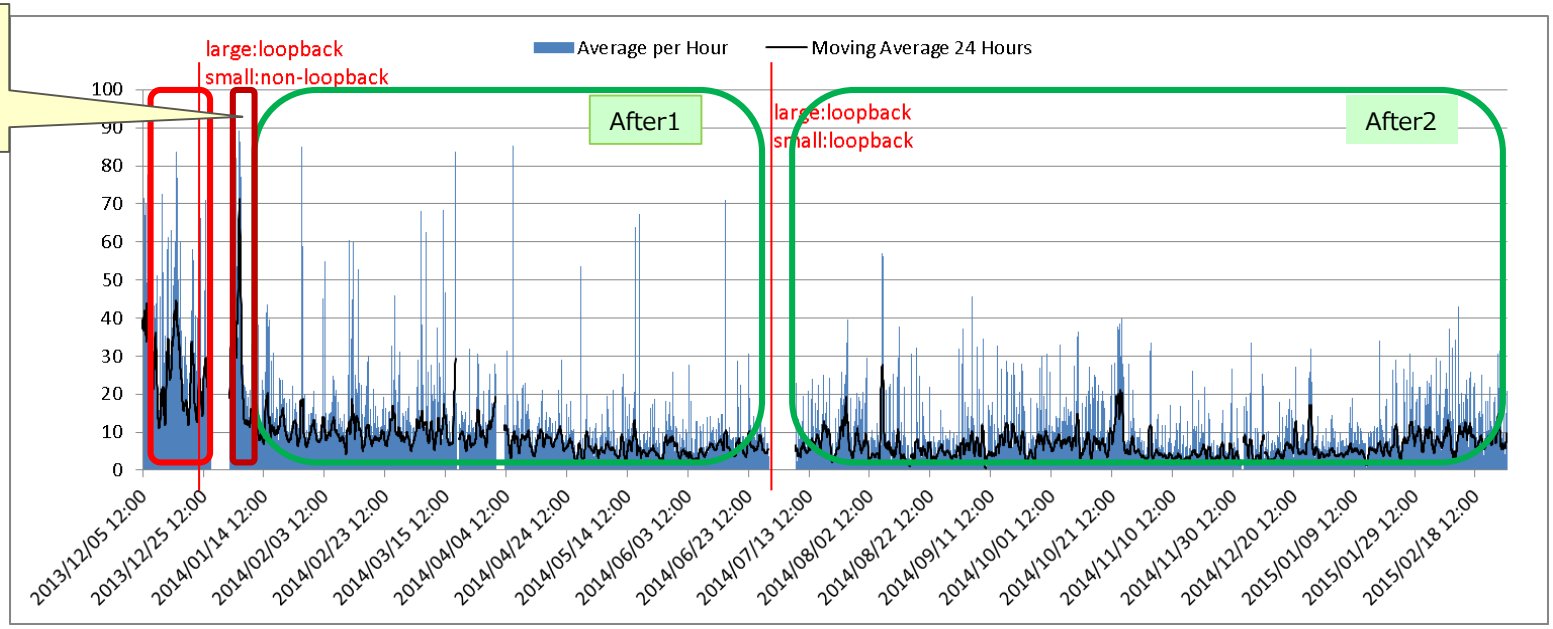
- Loopback Based Local FS Dramatically Scales over 9,000 Nodes!
  - Create 26K ops/node, unlink 37K ops/node by mdtest 100 files/node
  - Providing higher constant meta data access performance for each node



MDS CPU Load Comparison (Before vs. After)
Longtime evaluation except maintenance time(2013/12-2015/2)

- MDS Load average per hour: about 1/3.5
- Peak occurrence times per day(Over 50%,70%): less than 1/30

Some Large Class Jobs did not apply Rank Local File System



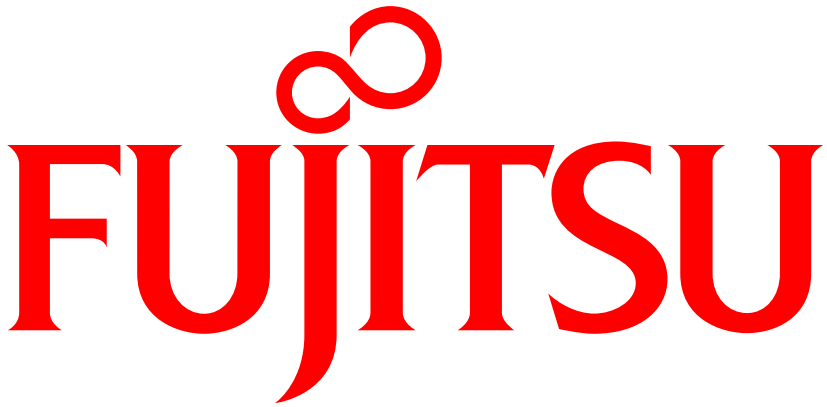
Need to add some option to use loopback based rank directory

	Before -13/12/27	After1: -14/6/29	After2: -15/2/28	After (All)
Average MDS Load %	25.1	8.21	6.36	7.13
Over 50% times per day	2.32	0.12	0.02	0.06
Over 70% Times per day	0.68	0.04	0.00	0.02

Summary

FUJITSU

- Meta Data Access Distribution by Loopback File System
  - Distributing and leveraging Meta Data and Data Access
  - Providing higher constant access performance on rank local file
- Evaluation
  - Achieved Better File Access Performance up to 128MB
  - Loopback Based Local FS Dramatically Scales over 9,000 Nodes!
  - MDS Load average: about 1/3.5
  - Peak Occurrence Times per Day: less than 1/30
- Introduction of Loopback based rank local file system is very effective on K computer operation
  even if 1 MDS+ 2,592 OSSes (5,184 OSTs) file system.



shaping tomorrow with you