Lustre & Cluster

- monitoring the whole thing -

Erich Focht
NEC HPC Europe

LAD 2014, Reims, September 22-23, 2014

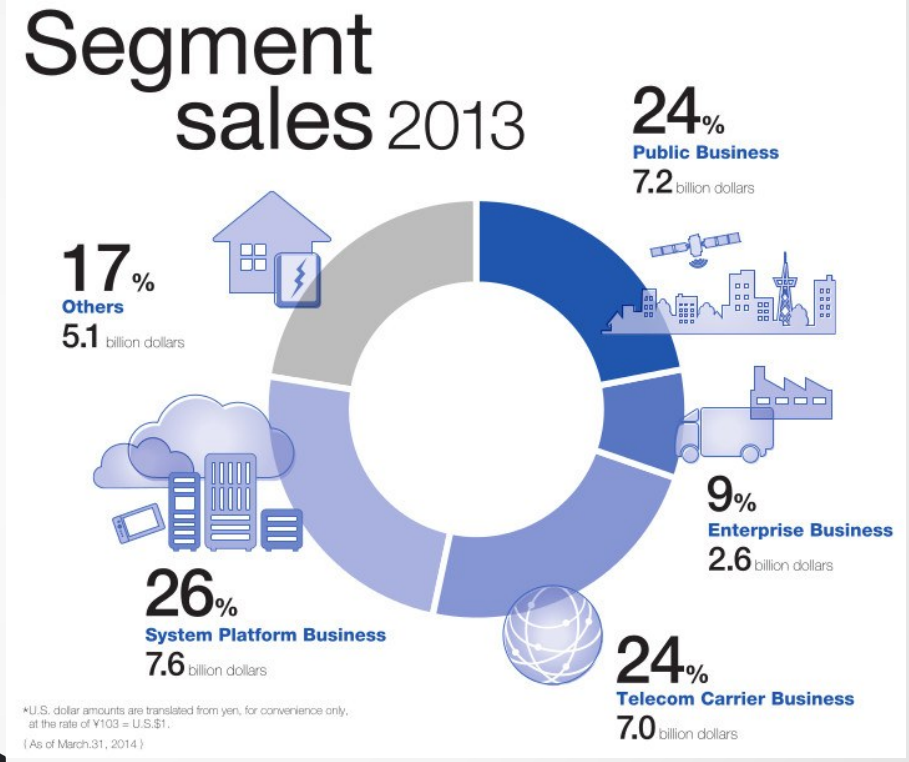
NEC

Overview

- Introduction
- LXFS
- Lustre in a Data Center
  - IBviz: Infiniband Fabric visualization
- Monitoring
  - Lustre & other metrics
  - Aggregators
  - Percentiles
  - Jobs

Introduction

- NEC Corporation
  - HQ in Tokyo, Japan
  - Established 1899
  - 100000 employees
  - Business activities in over 140 countries
- NEC HPC Europe
  - Part of NEC Deutschland
  - Offices in Düsseldorf, Paris, Stuttgart.
  - Scalar clusters
  - Parallel FS and Storage
  - Vector systems
  - Solutions, Services, R&D



Parallel Filesystem Product: LXFS

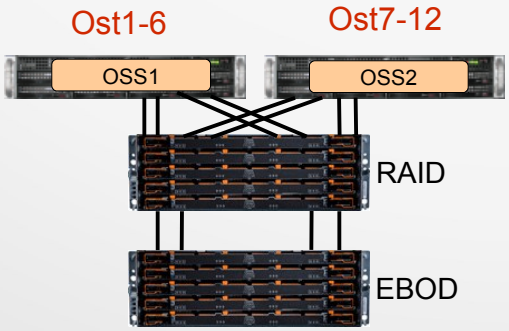
- Built with Lustre
- Since 2007: parallel filesystem solution integrating validated HW (servers, networks, high performance storage), deployment, management, monitoring SW.
- Two flavours:
  - LXFS standard
    - Community edition Lustre
  - LXFS Enterprise
    - Intel(R) Enterprise Edition for Lustre* Software
    - Level 3 support from Intel

Lustre Based LXFS Appliance

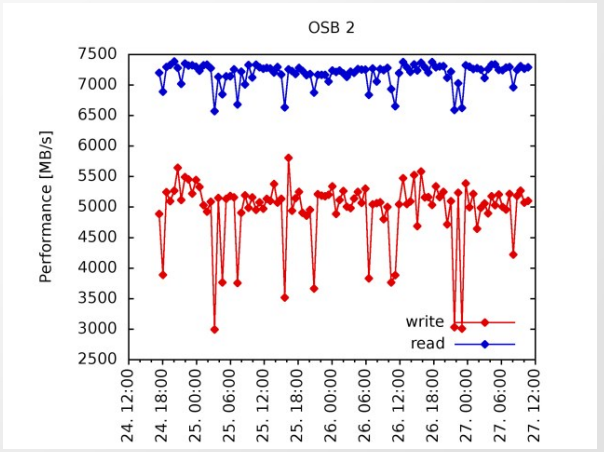
- **Each LXFS block is a pre-installed appliance**
  - Simple and reliable provisioning
- **Entire storage cluster configuration is in one place: LxDaemon**
  - Describes the setup, hierarchy
  - Contains all configurable parameters
  - ***Single data source*** used for deployment, management, monitoring
- **Deploying the storage cluster is very easy and reliable:**
  - Auto-generate configuration of components from data in LxDaemon
  - Storage devices setup, configuration
  - Servers setup, configuration
  - HA setup, services, dependencies, STONITH
  - Formatting, Lustre mounting
- **Managing & monitoring LXFS**
  - Simple CLI, hiding complexity of underlying setup
  - Health monitoring: pro-active, sends messages to admins, reacts to issues
  - Performance monitoring: time-history, GUI, discover issues and bottlenecks

Eg.: Performance Optimized OSS Block

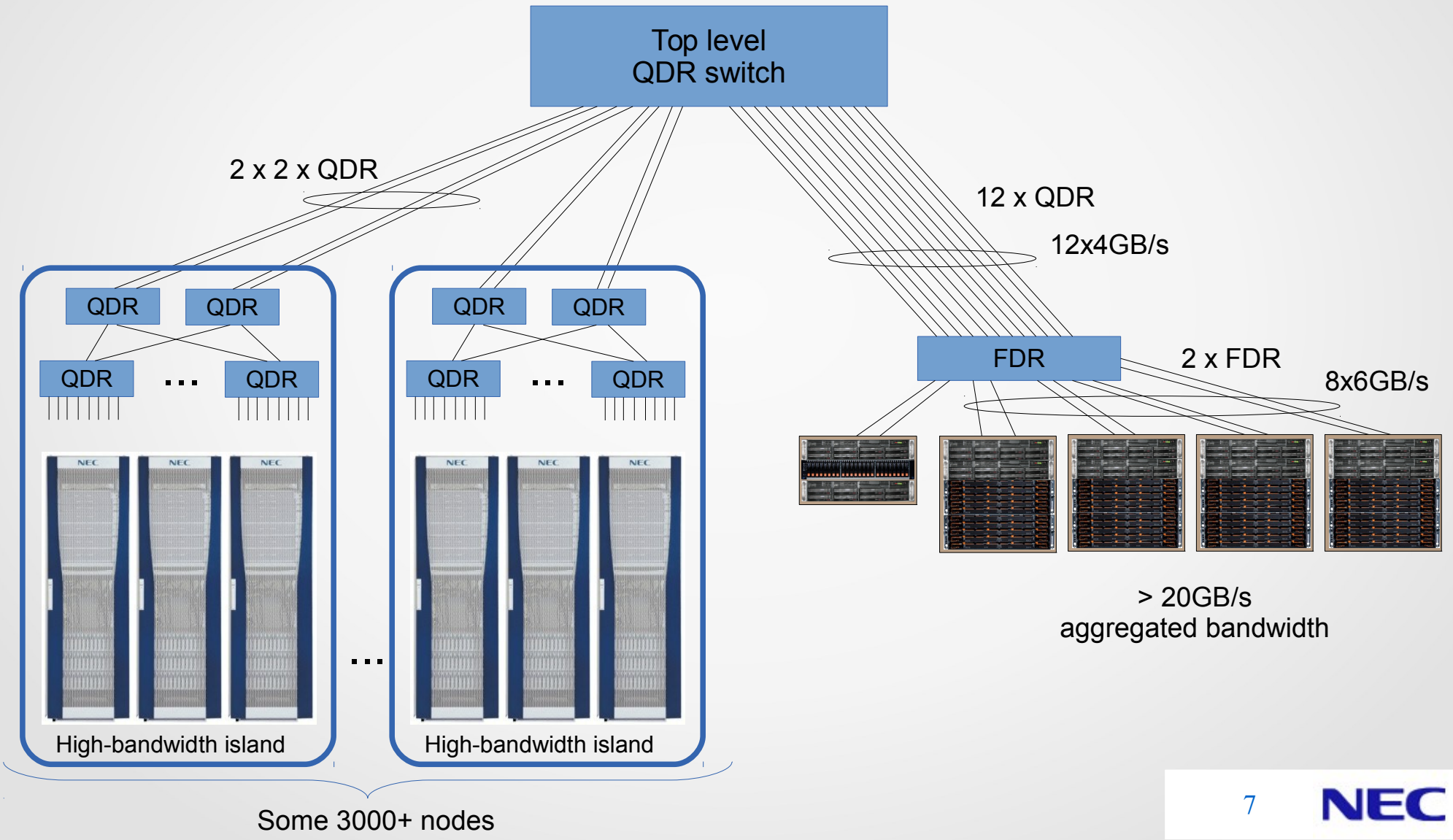
- Specs
  - > 6GB/s write, > 7.5GB/s read, up to 384TB in 8U (storage)
  - 2 Servers: highly available, active-active
    - 2 x 6core E5-2620(-v2)
    - 2 x LSI9207 HBA, 1 x Mellanox ConnectX 3 FDR HCA
  - Storage
    - NEC SNA460 + SNA060 (built on NetApp E5500)
- ...



Performance example
80 NL-SAS 7.2k
IOR
In noisy environment



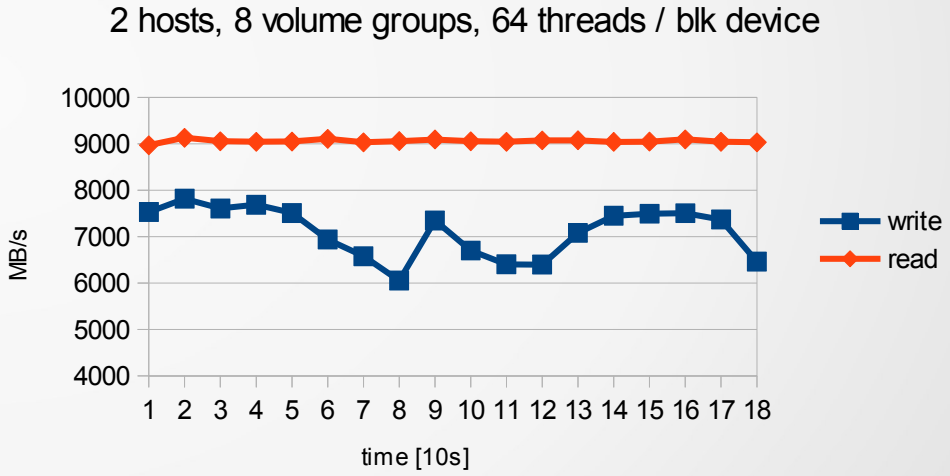
Concrete Setup: Automotive Company Datacenter



NEC

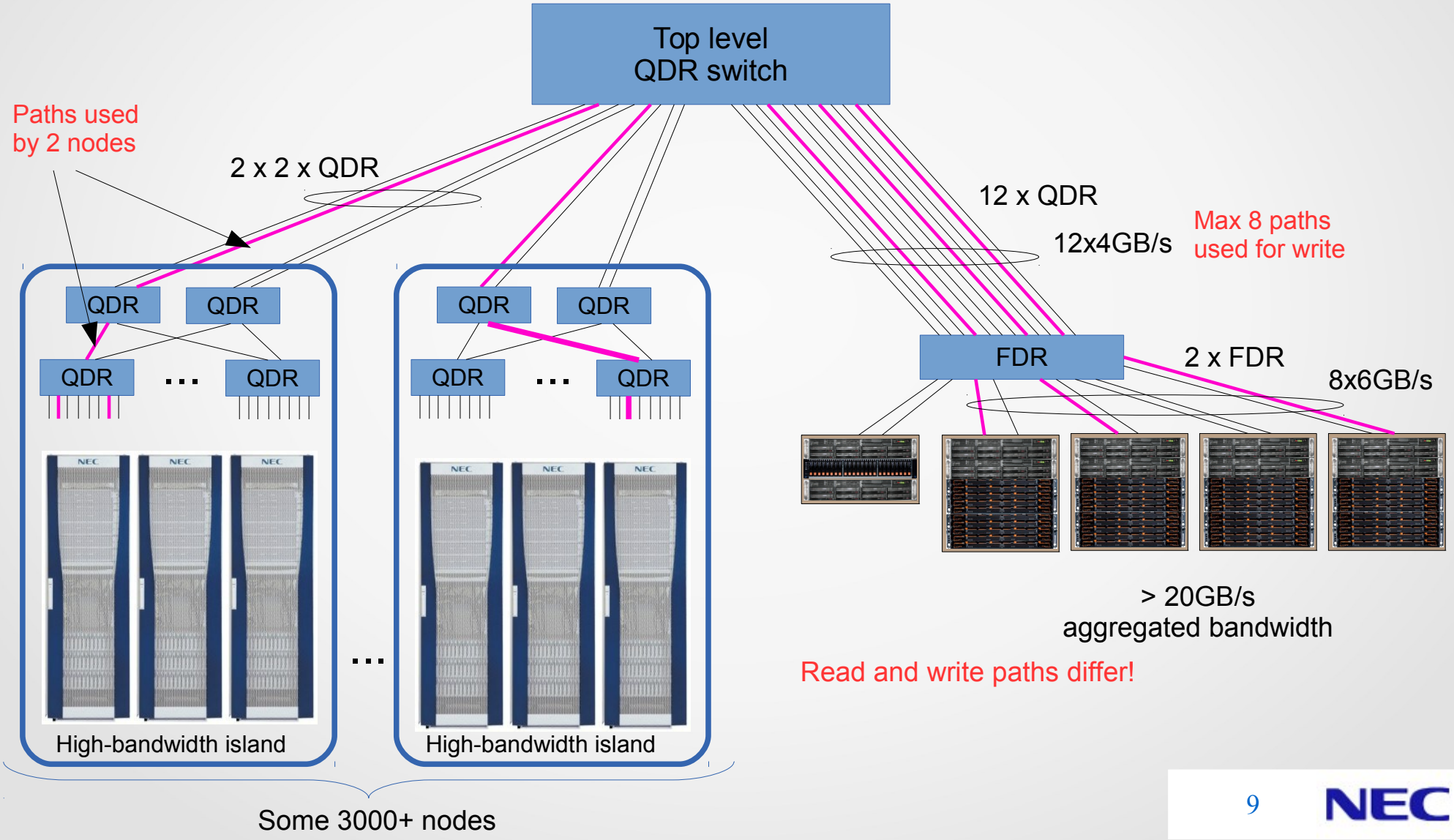
Demonstrate 20GB/s on 8 or 16 Nodes!

- Check all layers!
- OSS to Block Devices
  - vdbench, sgpdd-survey, with Lustre: obdfilter-survey
  - Read/write to controller cache! Check PCIe, HBAs (SAS)
- LNET
  - Lnet-selftest
  - Peer credits? Enough transactions in flight?
- Lustre client
- Benchmark
  - IOR, stonewalling
- ... but ...



NEC

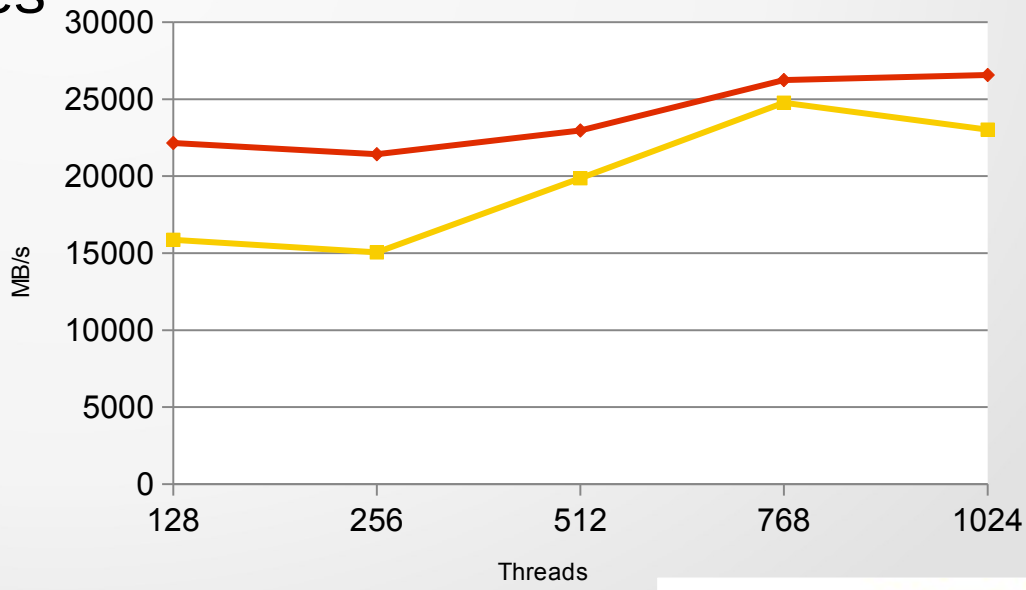
The Infiniband Fabric: Routes!



NEC

Demonstrate 20GB/s on 8 or 16 Nodes!

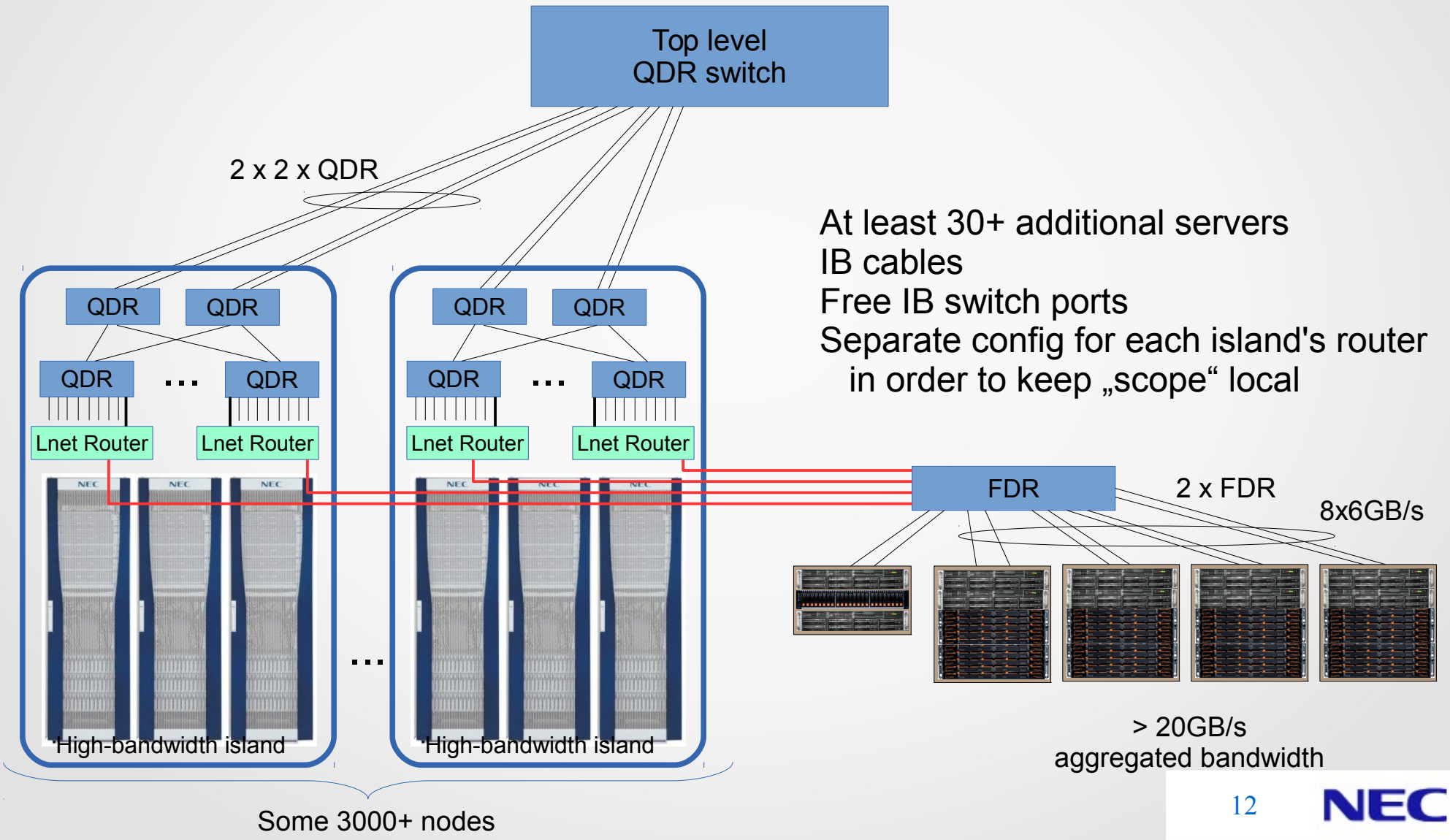
- Given the routing is static: select nodes carefully
  - Out of a pool of reserved nodes
  - Minimize oversubscription of InfiniBand links on both paths
  - ... to OSTs/OSSes that are involved
  - ... and back to client nodes
- Use stonewalling
- Result from 16 clients
  - Write: max 24.7GB/s
  - Read: max 26.6GB/s



Trivial Remarks

- Performance of storage devices is huge
  - Performance of one OSS is O(IB link bandwidth)
  - If overcommitting inter-switch-links we hit quickly the limit
- Sane way of integrating is more expensive and more complex:
  - Lustre/Lnet routers
    - Additional hardware
    - Each responsible for its „island“
    - Can also limit I/O bandwidth in an „island“

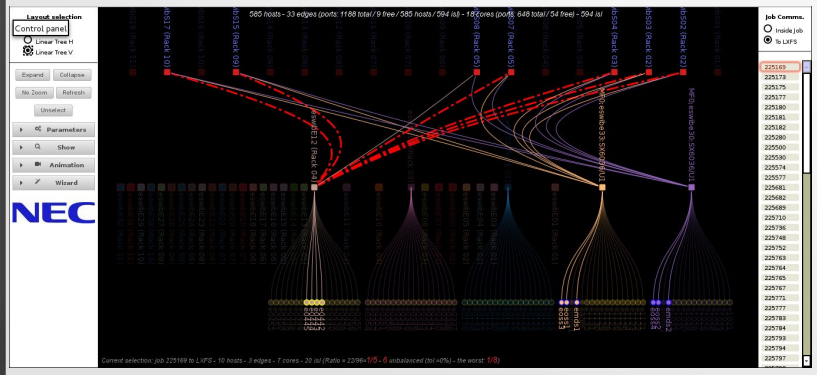
Integrated In Datacenter: The Right Way

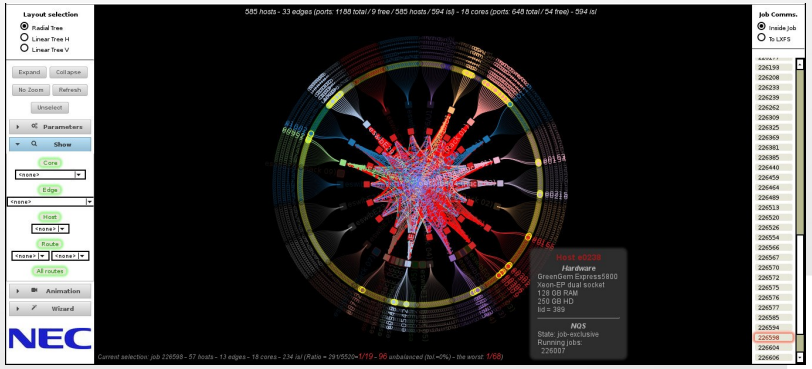


At least 30+ additional servers
IB cables
Free IB switch ports
Separate config for each island's router
in order to keep „scope“ local

IBviz

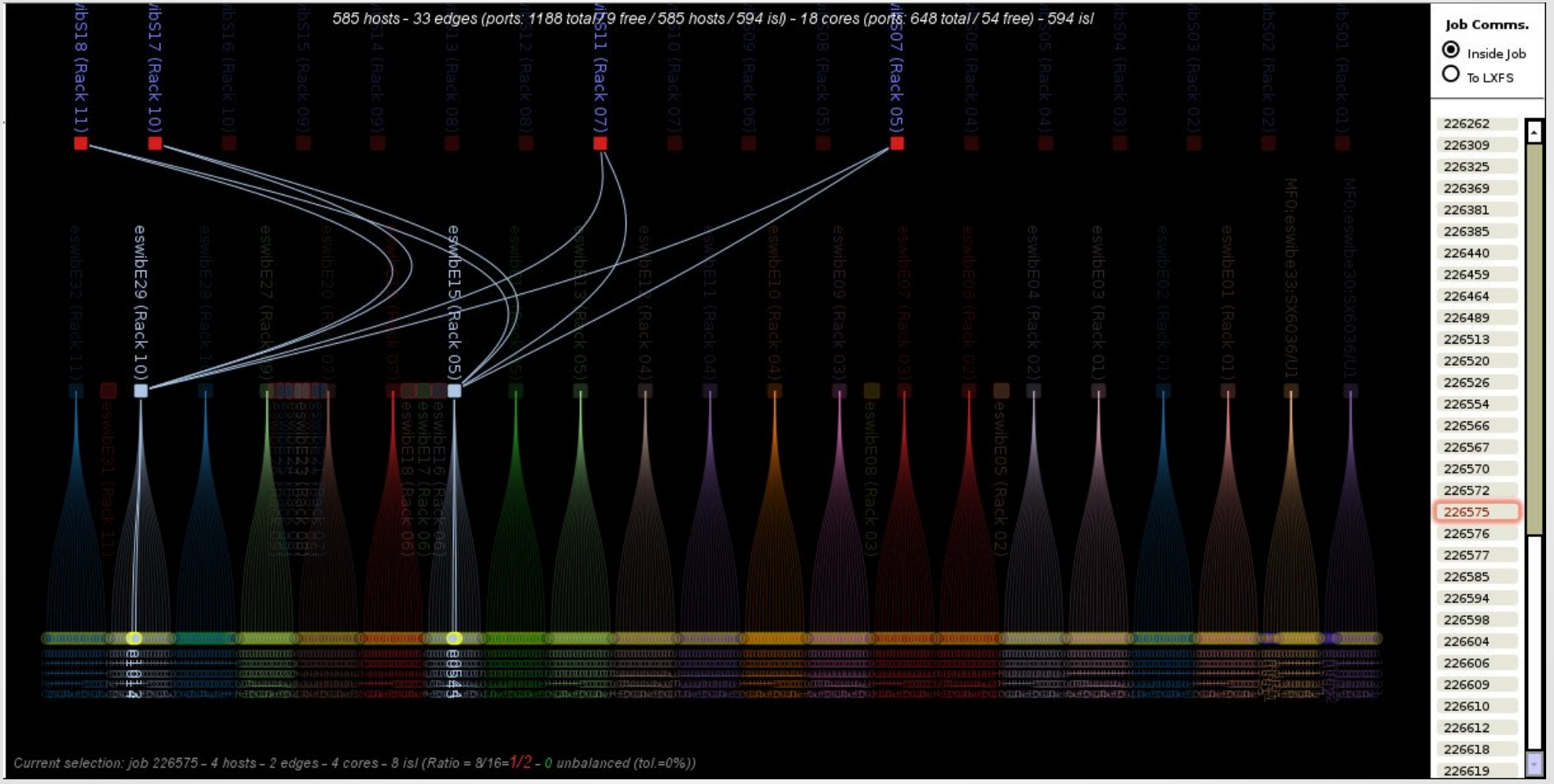
- Development inspired by benchmarking experience described before
- Visualize the detected IB fabric, focus on inter-switch links (ISLs)
  - Mistakes in network are ***visible*** (eg. missing links)
- Compute number of static routes (read/write) going over ISLs
  - Which links are actually used in an MPI job?
  - How much bandwidth is really available?
  - Which Links are used for IO and what bandwidth is available)





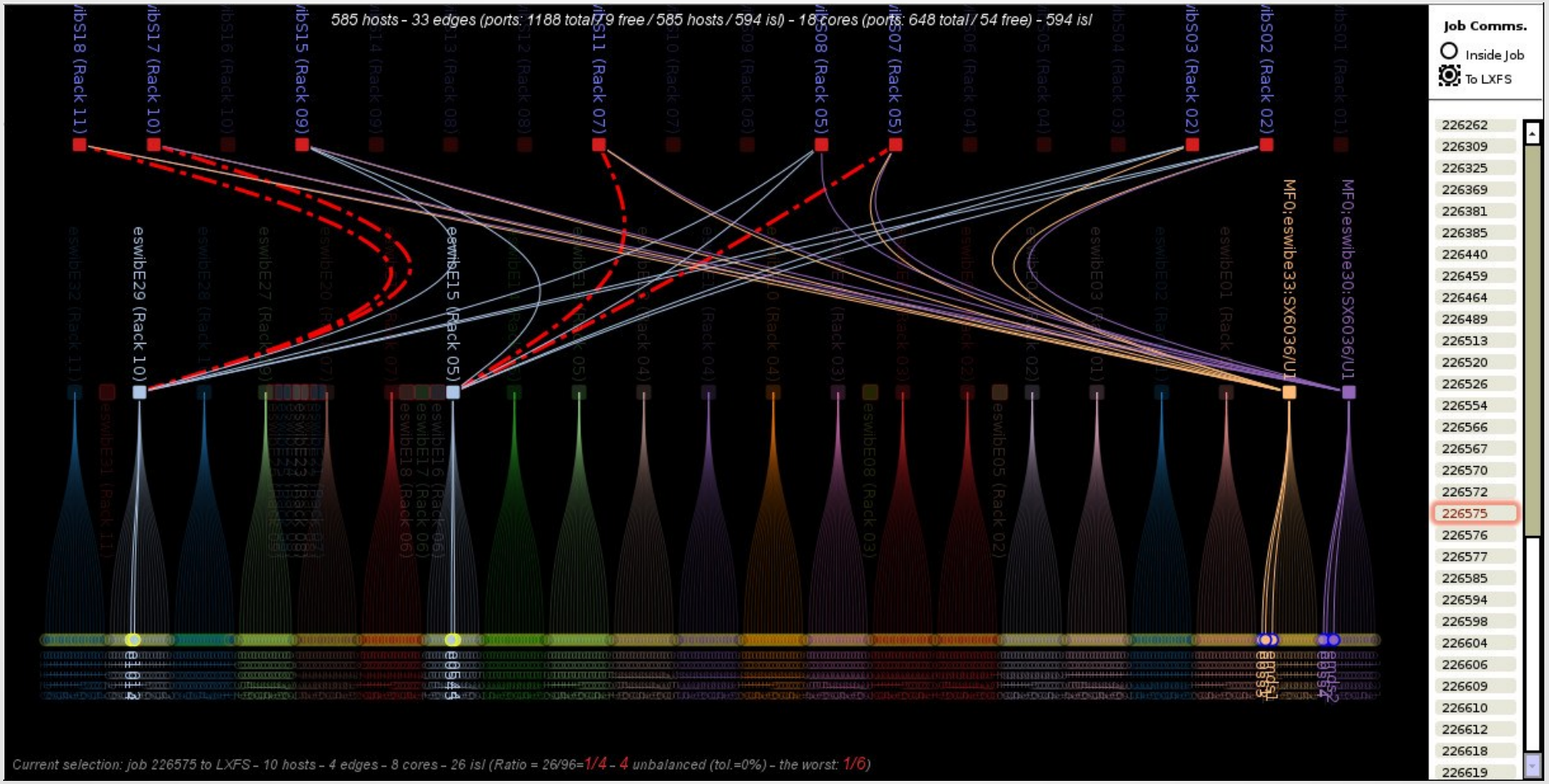
NEC

Routes Between Compute Nodes: Balanced



NEC

Routes to Lustre Servers: Unbalanced



NEC

Monitoring Cluster & Lustre Together

- Lustre Servers ... ok
  - Ganglia standard metrics
  - own metric generators delivering metrics into Ganglia, Nagios
- Cluster Compute Nodes ... ok
  - See above ... Ganglia, Nagios, own metrics
- IB ports ... ok
  - Own metrics, in Ganglia, attached to hosts
  - Need more and better
- IB routes and inter-switch-link stats ...
  - Routes: ok, ISL stats: doable
  - Don't fit well into hierarchy... Ganglia won't work any more

Monitoring Cluster & Lustre Together

- Lustre Clients ...
  - Plenty of statistics on clients and servers
  - often: no access to clients: use metrics on servers
  - Scaling problem!
    - At least 4 metrics per OST and client
    - 1, O(10), more metrics per MDT and client
    - 1000 clients, 50 OSTs: 200k additional metrics in monitoring system
- Jobs
  - Ultimately, we want to know which jobs are performing poorly on Lustre, why, advise users, tune their code, the filesystem, use ost pools...
  - Either map client nodes to jobs (easy, no change on cluster needed)
  - Or use (newer) tagging mechanism provided by Lustre
  - Store metrics per job!? Rather unusual in normal monitoring systems

Monitoring Cluster & Lustre Together

- Aggregation!
  - Reduce number of metrics stored
  - Do we care about each OST read/write bandwidth for each client node?
    - Rather look at total bandwidths and transaction sizes of node
    - Example: 50 OSTs * 4 metrics -> 4 metrics
      - 200k metrics -> 4k metrics to store
      - But still processing 200k metrics
  - What about jobs?
    - In order to see whether anything goes wrong, do we need info on each compute node?

The work described on the following pages has been funded by the German Ministry of Education and Research (BMBF) within the FEPA project.

Aggregate Job Metrics: in FEPA Project

- Percentiles
  - Instead of looking at the metric of each node in a job, look at all at once

Example data:

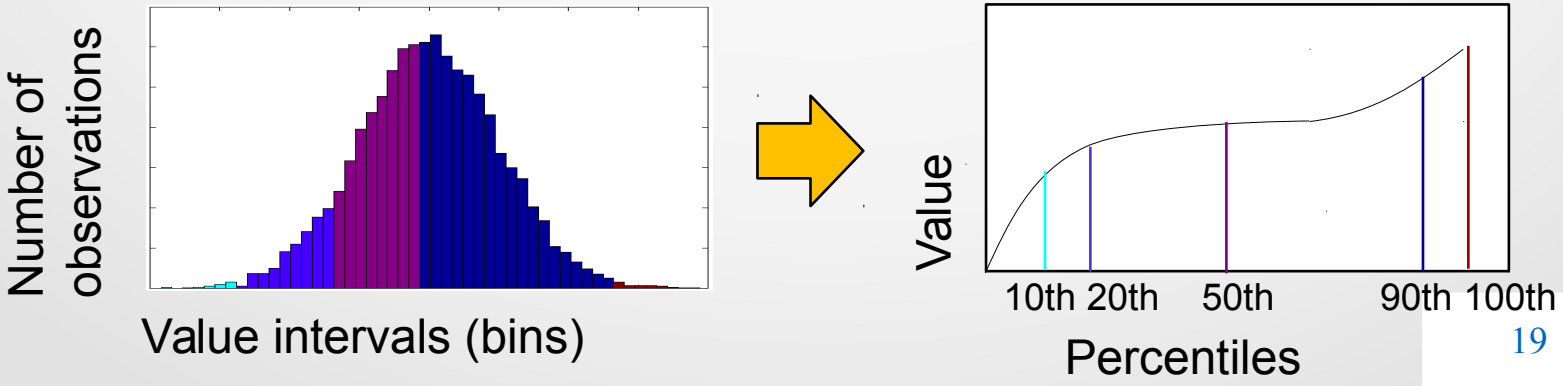
7	8	10	7	6	3	6	7	12	8	8	9	4	8	9	8	5	11	10	7

Sort your data:

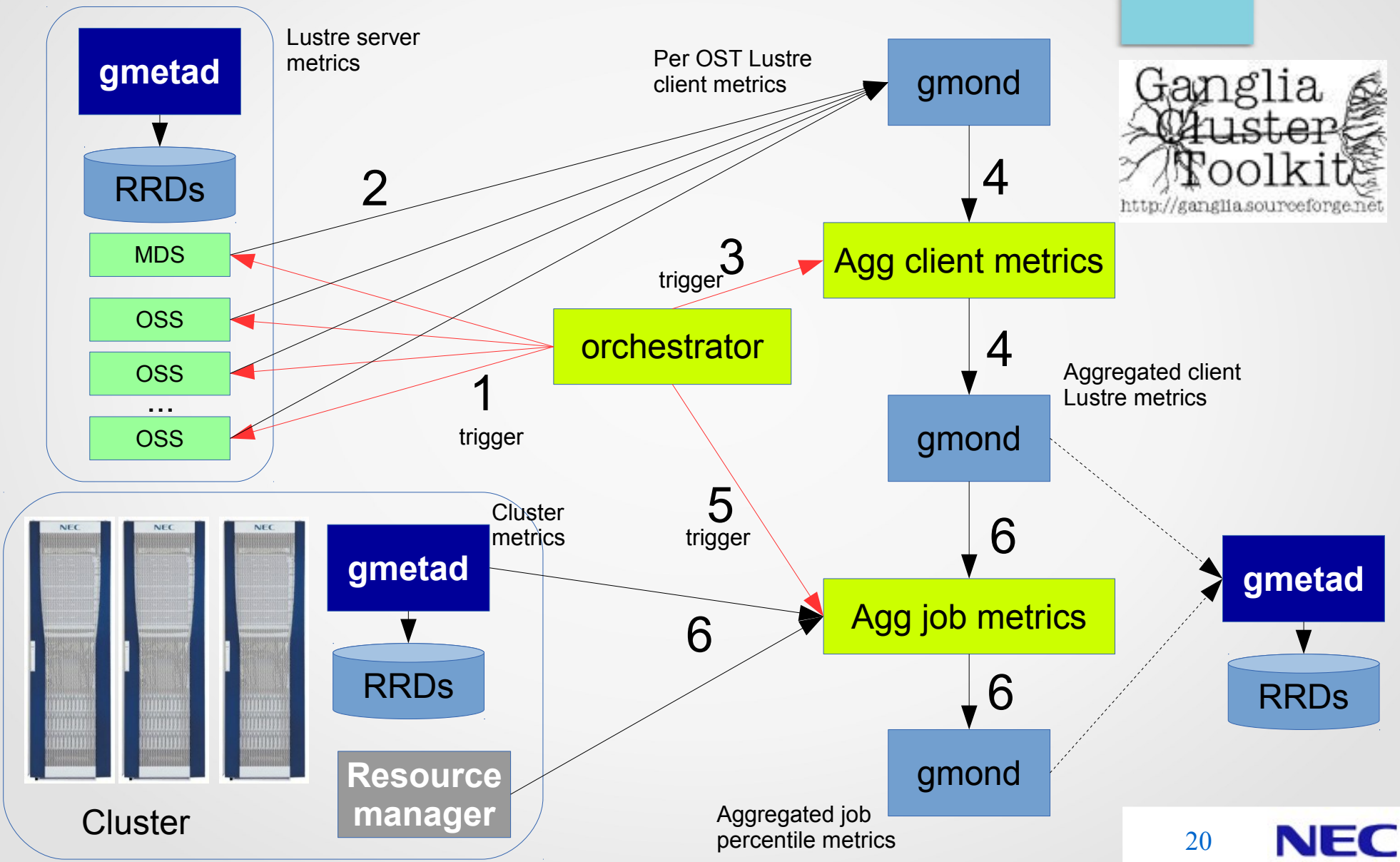
3	4	5	6	6	7	7	7	7	8	8	8	8	8	9	9	10	10	11	12

Get the percentiles:

0th	10th		20th		30th		40th		50th		60th		70th		80th		90th		100th
3	4	5	6	6	7	7	7	7	8	8	8	8	8	9	9	10	10	11	12

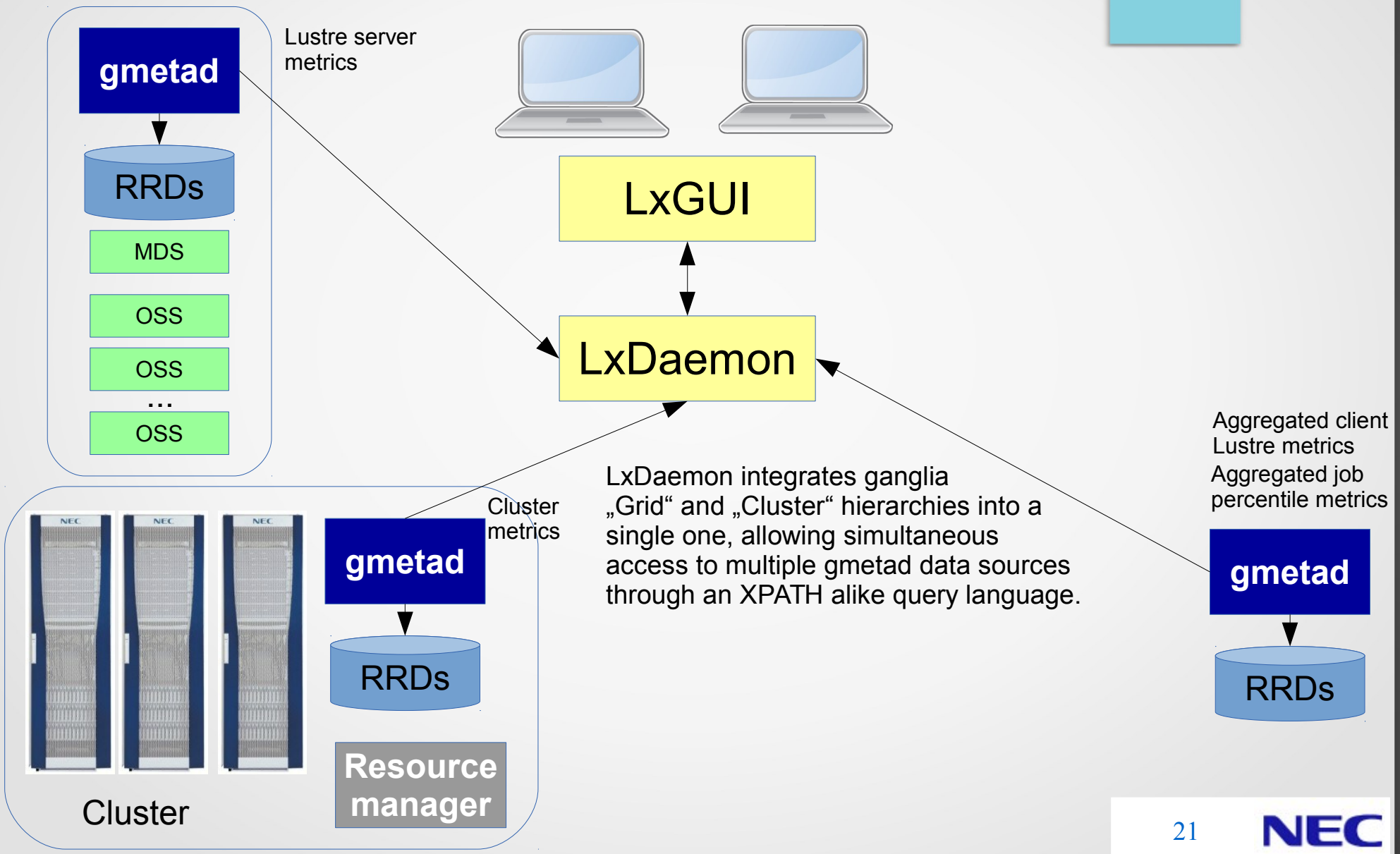


Architecture: Data Collection / Aggregation

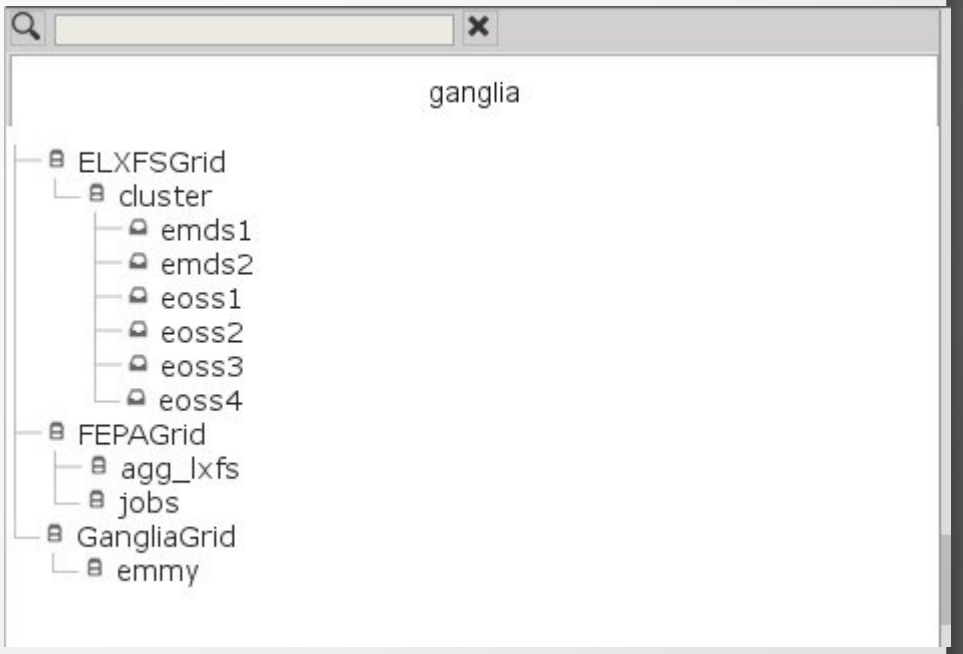


NEC

Architecture: Visualization

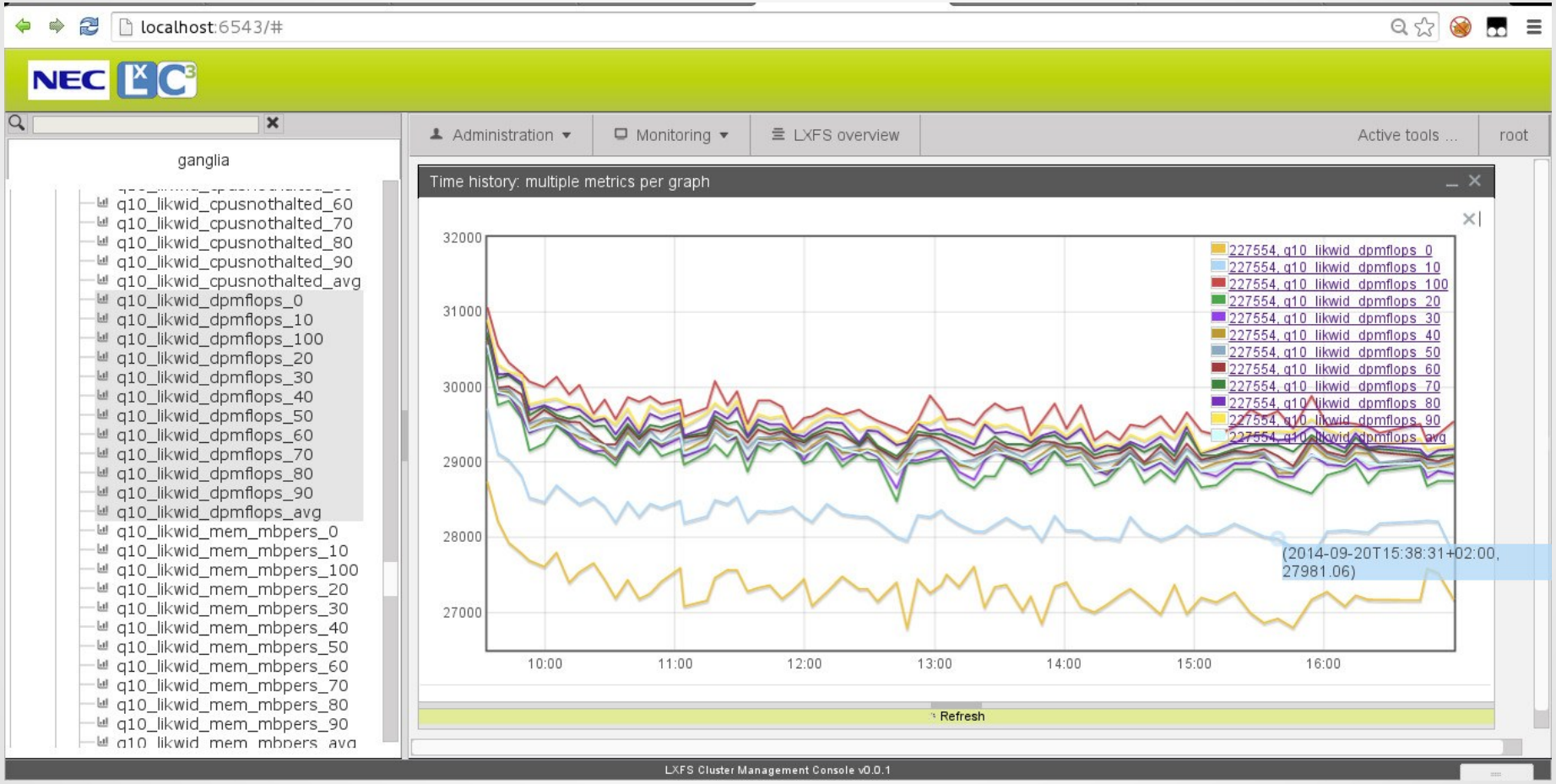


Metrics Hierarchies



NEC

Cluster Metrics: DP MFLOPS Percentiles

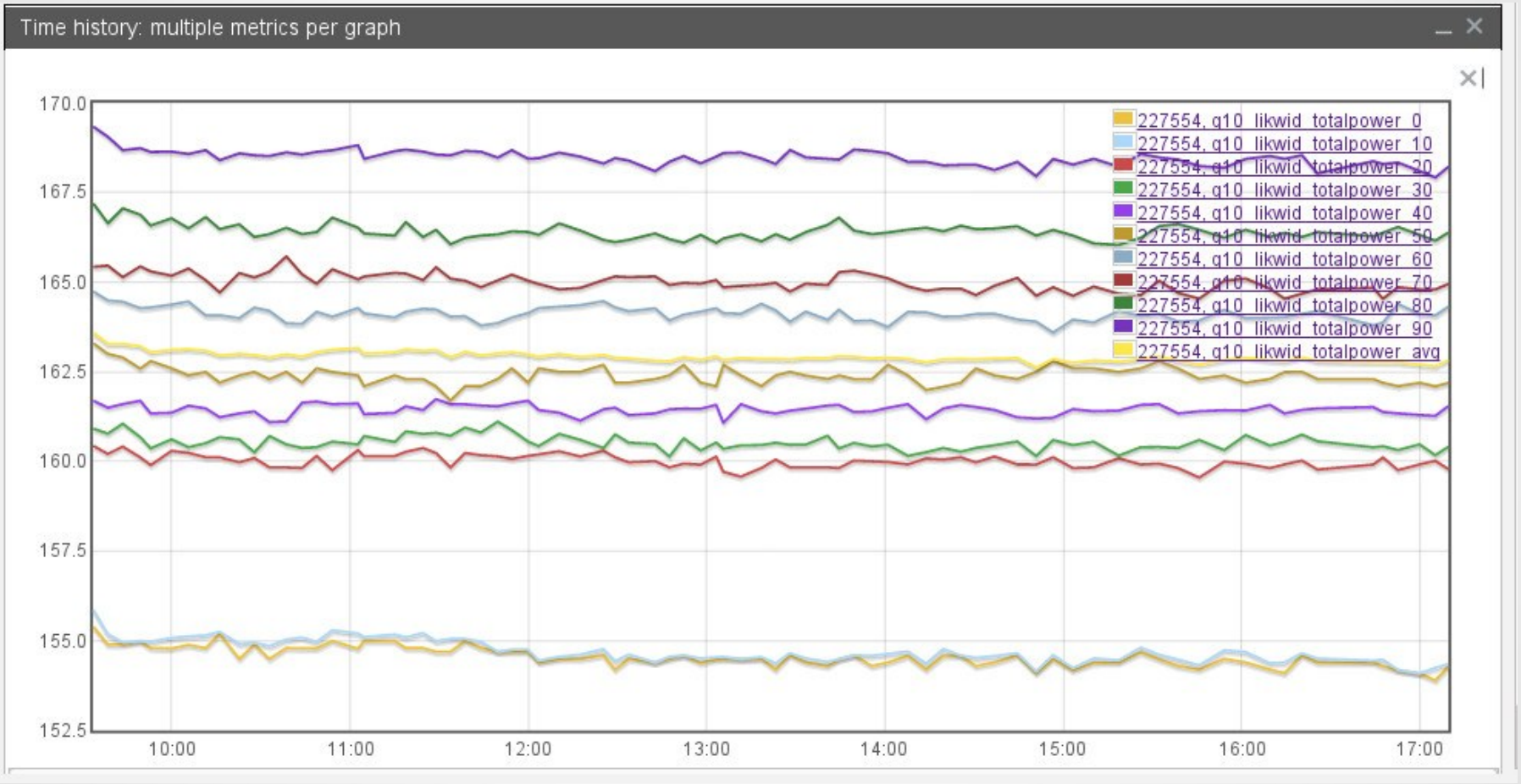


MFLOPS avg vs. Lustre Write Percentiles

Job: Lustre Read KB/s Percentiles

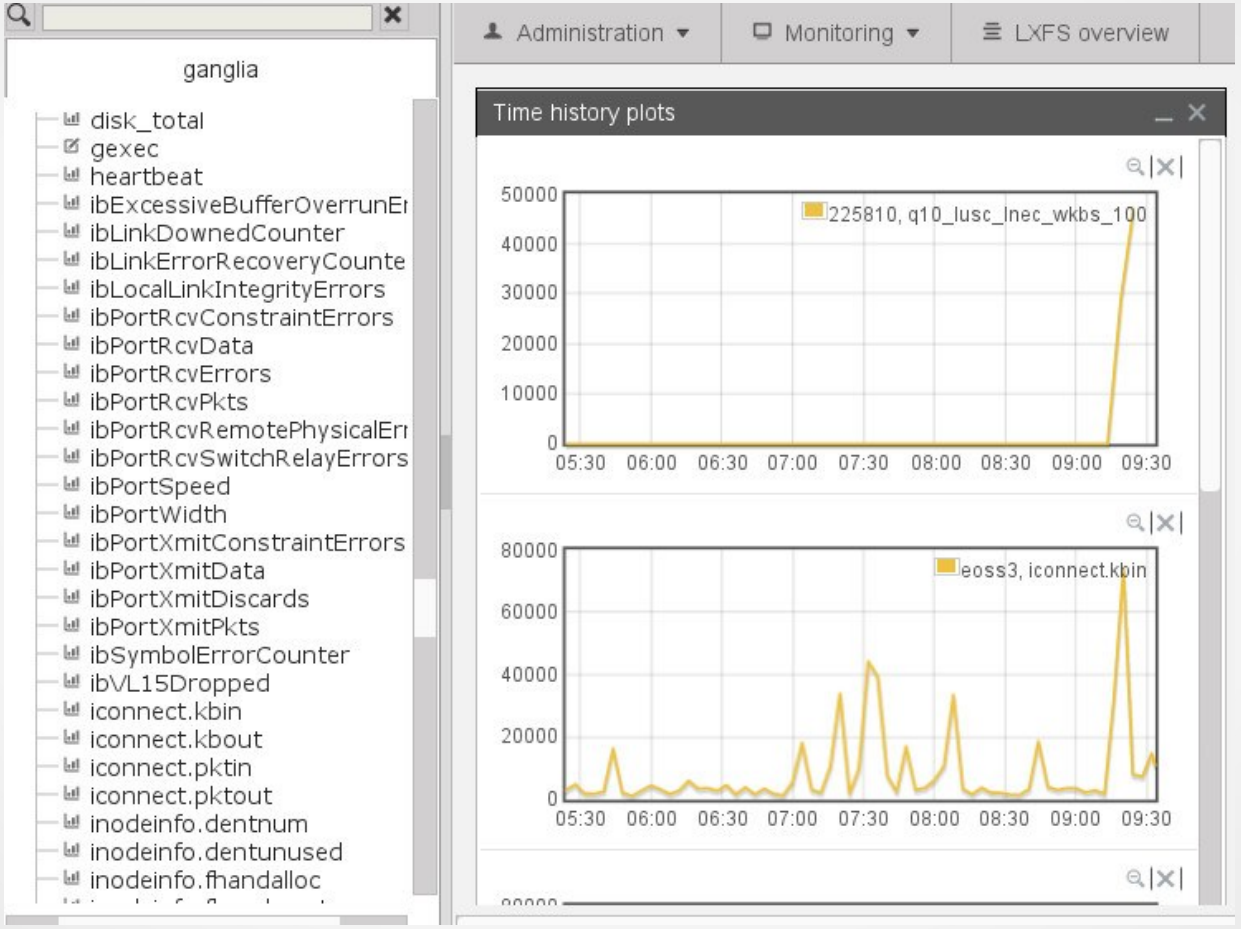
NEC

Totalpower Percentiles in a Job



NEC

Metrics from different Ganglia Hierarchies



NEC

Conclusion

- Computer systems are complex:
- Infiniband network: even static metrics help understand system performance
- There are plenty of metrics to measure and watch
- Storing all of them is currently not an option
  - Aggregation
- Per Job metrics are most useful for users.
  - But monitoring systems not really prepared for jobs data
- Ganglia turned out to be quite flexible and helpful, but gmetad and RRD have their issues...