« *Last night a BeeG$_{FS}$ saved my life* »

Romaric DAVID
Michel RINGENBACH
david@unistra.fr, mir@unistra.fr
Direction Informatique

Octobre 2016

UNIVERSITÉ DE STRASBOURG

Plan

- Context
- Storage design options
- 3 differents architectures with BeeGFS
- Conclusion

Forewords

- This presentation is a feedback on 2 years of deployment of filesystems at the HPC centre of University of Strasbourg
- We all have to cope with Big Data: volume, variability, velocity, ...
- Users want the filesystem to work just as in the good old times, no matter the underlying technology
- New technology all the time, which changes should we consider ?

Forewords

- See *Succes 2015* and *Jres 2015* presentations
- Q4 2015 we started migrating to RozoFS for our home directories.
  We bought the H/W (standard Dell R730 / MD1400), We bought the S/W
- Q2 2016 : Too many bugs, some annoying blocking ones ⇒ We had to move away from Rozo

High Performance Computing Center : Fundings

The HPC Center of the Unistra is funded by:

- Unistra: hosts the engineers responsible for the HPC Center
- The research labs fundings: until 2013, 100% of compute servers had been bought by the labs
  Labs are located not only in Strasbourg, but in all the Alsace region (too many logos to show)
- The French national initiative *Investissements d'Avenir*, *via* a national project: Equip@Meso
- French government, Alsace Region and Strasbourg Eurométropole

HPC in Strasbourg: Flops, Bytes....

Last night a BeeGFS saved my life

October 2016

- Around 400 servers, 5500 cores
- 450 TB of GPFS Storage (on departure)
- 1 PB of BeeGFS storage (being installed)
- Many TB of BeeGFS for scratch
- 60 GPUs, from Tesla
  M2050 to K80
- 270 Tflops
- More than 250 active users
- More than 150 softwaremodules

HPC in Strasbourg - Context

- Regarding data, We need to face the following challenges:
  - Growing number of users ⇒ Growing numbers of use cases ⇒ impact on filesystem
  - Growing amount of Data ⇒ Not just more, but differently
  - Scope of mutualization of computing ressources ⇒ users want their own storage units on our site
- Translated into features: scalable, modular, cheap, smart...

Plan

- Context
- Storage design options
- 3 differents architectures
- Conclusion

Storage design options

- Our old GPFS on Infiniband was not that performant and could not evolve easiliy ⇒ why use IB for storage?
- Now our compute nodes embed 10 GbE Interfaces ⇒ Let's use these interfaces for access to storage
- Omnipath is coming...
- Let's split networks : 10GbE for Storage, IB or OPA for compute
- For storage itself, we only want to use 2 SSD and 7200 RPM capacitive drives

Storage design options

- Life is parallel: one important point is *agregated bandwidth* to storage, not point-to-point bandwidth
- Agregated bandwith (scale-out) will be reached by adding servers. 10GbE network not that expensive. No more SAN!
- Scale-up will be reached by adding disks to servers
- As we need users to be able to add disk space from time to time at low cost ⇒ standard H/W
- We are OK to deal with differents namespaces (/home, /scratch....)

Storage design options

- We need smart storage ⇒ SDS: GPFS, GlusterFS, HDFS, BeeGFS (ex Fraunhofer FS)
- Benefits:
  - Cost-optimized
  - Network for home file access independant from inter-node communication network
  - Standard GbE used at its maximum (whereas nearly not used on previous architectures)
  - We benefit from new node design from Dell
- We used BeeGFS for 3 different purposes on 3 different H/W setups (no, that's not 9 combinations)

Plan

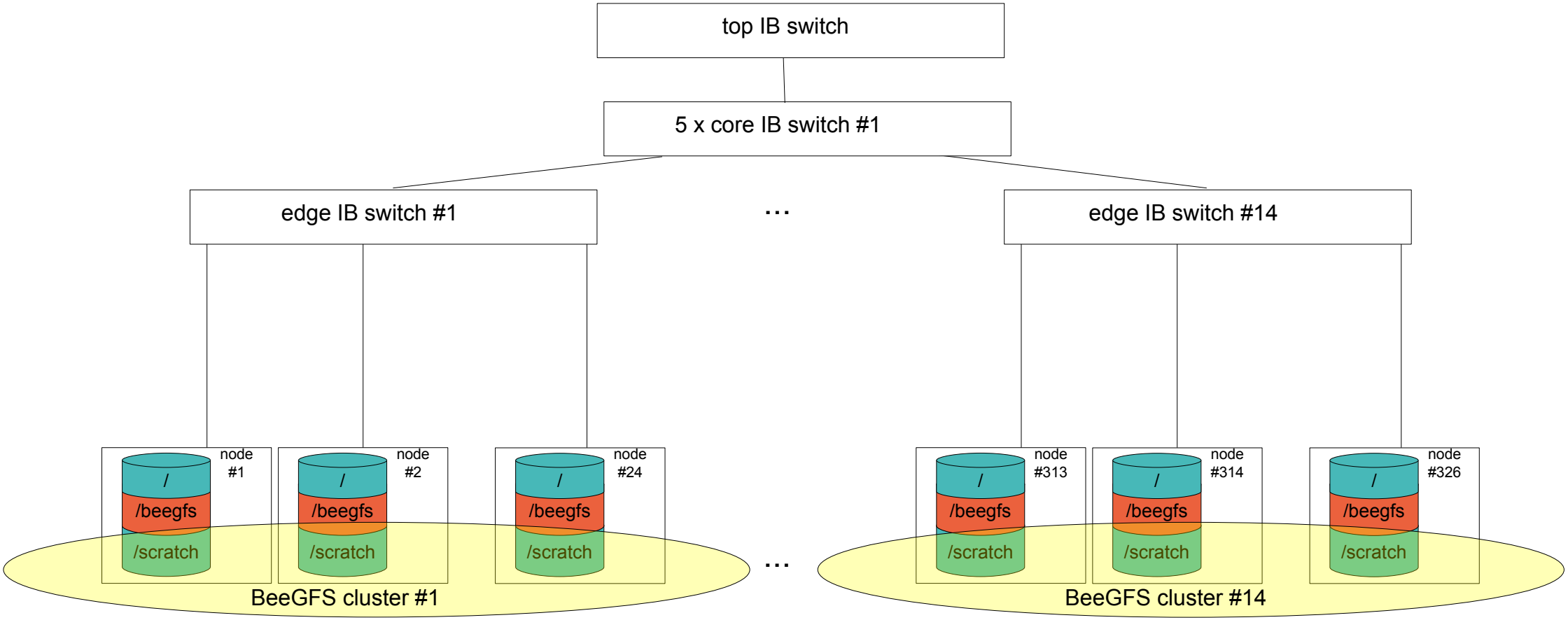
- Context
- Storage design options
- 3 different architectures
- Conclusion

3 different architectures

- A- Cost-optimized scratch
  - Idea: During jobs, let's use local disks nodes as agregated scratch
  - Features: no redundancy, no Raid, High-Sped-Network
- B- Temporary home directory while transferring data from one FS to another:
  - We are in need for a temporary reliable home
  - Features: no Raid, Mirroring, Ethernet network
- C- Home directory:
  - Features: RAID 6, Ethernet network, Mirroring

3 different architectures - A

Cost-optimized scratch:



3 different architectures - A

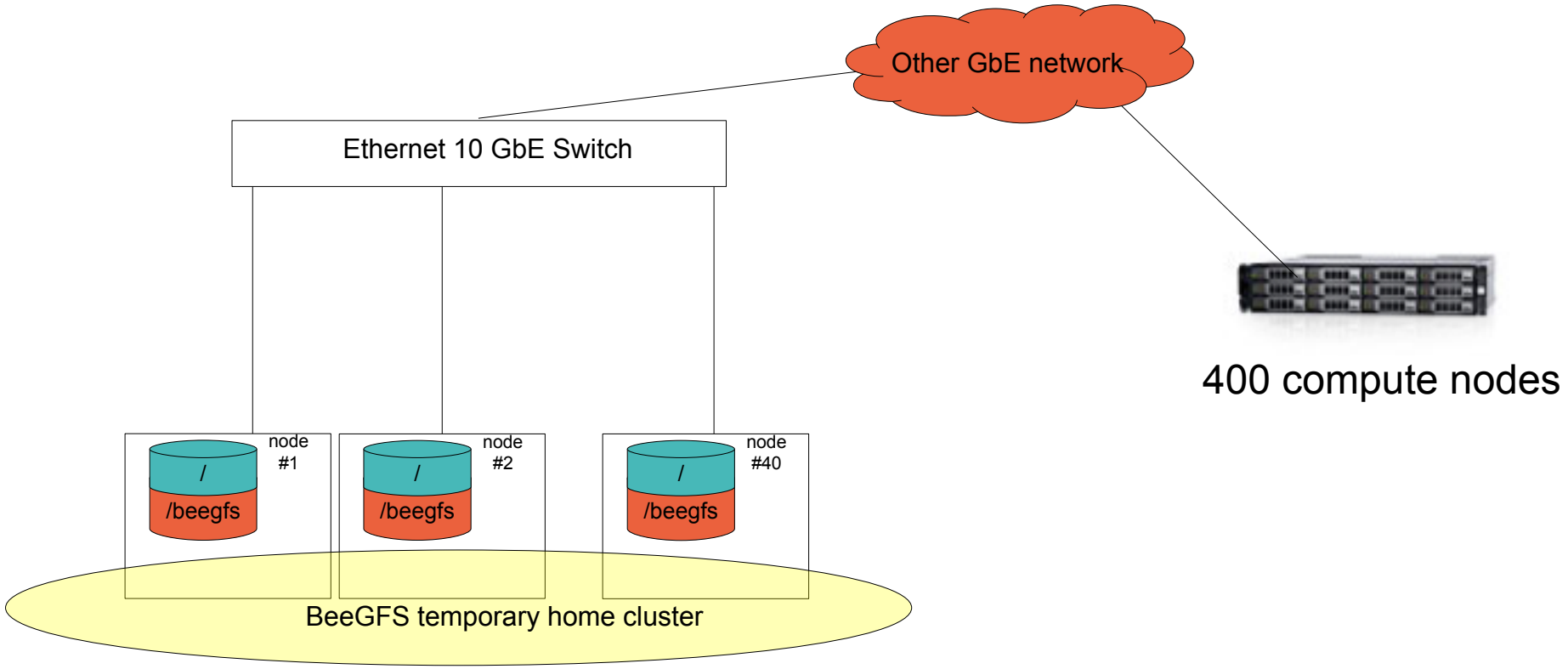
- In order to do that, please carefully read the documentation and... do the exact opposite:
  - No dedicated FS for BeeGFS *storage targets*
  - No dedicated Meta-Data Nodes
  - Meta-Data and data on 7200 RPM disks
  - Clients are also servers and comptue nodes
  - **0 € / TB**
- Anyhow:
  - Agregated Bandwidth up to 1.8 GB / s when 12 clients are doing *dd*
  - Easy set-up

3 different architectures - B

- Temporary home directories
  - Compared to architecture A, we don't use High-Speed network anymore, we use (buddy)-mirroring, on a 40-head storage cluster
  - Storage nodes are compute nodes, which are not yet computing....
  - No dedicated meta-data servers
  - Meta-Data and Data on 7200 RPM disks
  - Storage servers are only used as storage servers
- Scale-out OK, Scale-up limited to the number of disks in a server

3 different architectures - B

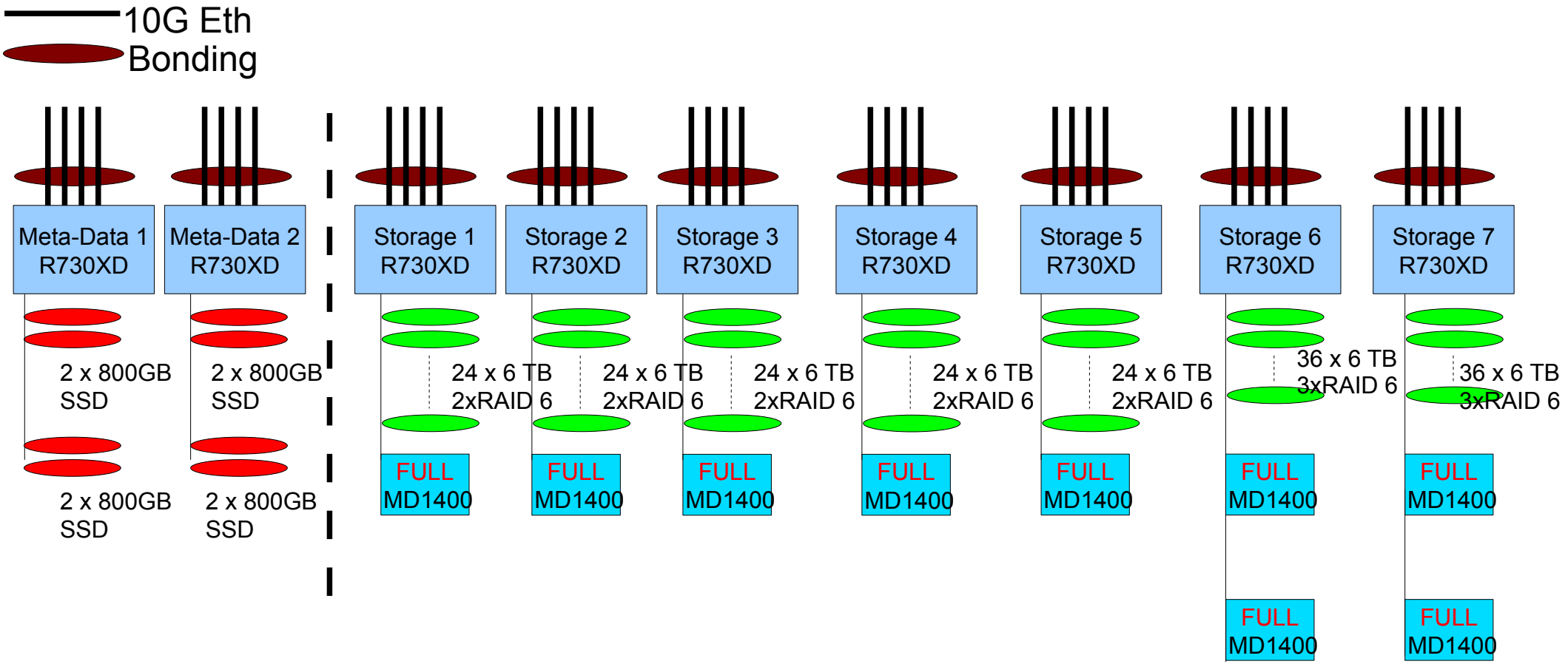
- Storage now relies on Ethernet



3 different architectures - C

- Permanent home directories
  - We add RAID arrays as storage targets
  - We use dedicated meta-data servers with SSD
  - Channel-bonding for Ethernet Interfaces (4 x 10Gb)
- Scalable (up and out)
- Safety features for home: Support, RAID, *Buddy* Mirroring, No snapshots yet, backup planned
- Target available space: around 2 PB but...
  - Scale out and up possible
- As storage is on Ethernet, we can switch to any high-speed network for HPC. IB → OPA

3 different architectures - C



Plan

- Context
- Storage design options
- 3 differents architectures with BeeGFS
- Conclusion

Conclusion

- Our strategy: build storage on commodity H/W
- Allows for incremental fundings and cost forecast
- Different architectures for different needs, thus different namespaces:
  - Scratch (close to compute, pseudo-local filesystem)
  - Home (shared distributed filesystem, average speed)
- Keep data close to compute...
  - Data staging home ↔ Scratch not automated (yet ?)
  - Users in need for performance are OK to deal with 2 namespaces

Conclusion

- Our strategy: build storage on commodity H/W
- Allows for incremental fundings and cost forecast
- Different architectures for different needs, thus different namespaces:
  - Scratch (close to compute, pseudo-local filesystem)
  - Home (shared distributed filesystem, average speed)
- Keep data close to compute...
  - Data staging home ↔ Scratch not automated (yet ?)
  - Users in need for performance are OK to deal with 2 namespaces