KERLABS

# Open Discussion
# File Systems

Renaud Lottiaux

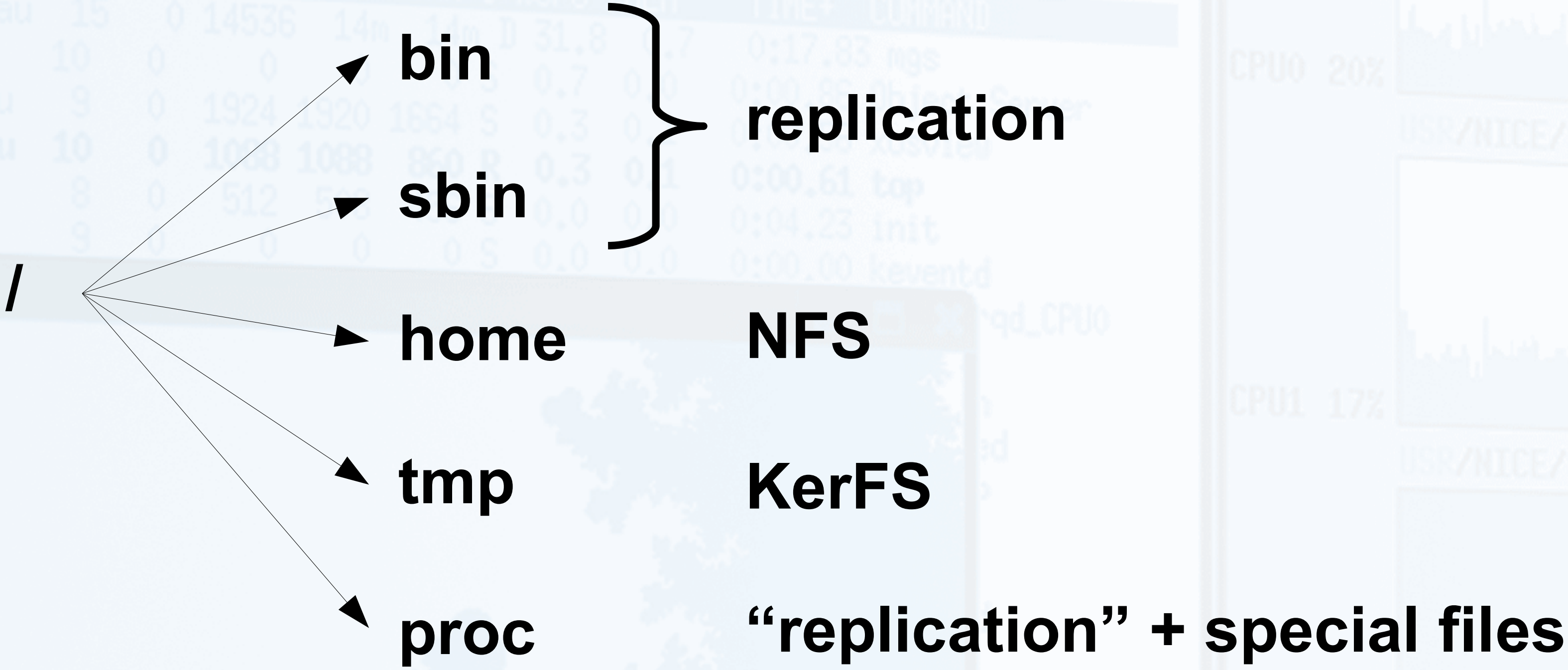
Kerrighed

Linux clusters made easy

# What is the File System issue in Kerrighed ?

- In Kerrighed processes can migrate
  - After a migration, processes must see same files and same file content
  - All processes must see same files and same content
- We (at least Kerlabs) need strong and stable solution
  - Customers would not accept unstable or experimental FS
  - Many customer will ask for a specific file system

# What have been done previously (V1.0.2)



/

bin
sbin
replication

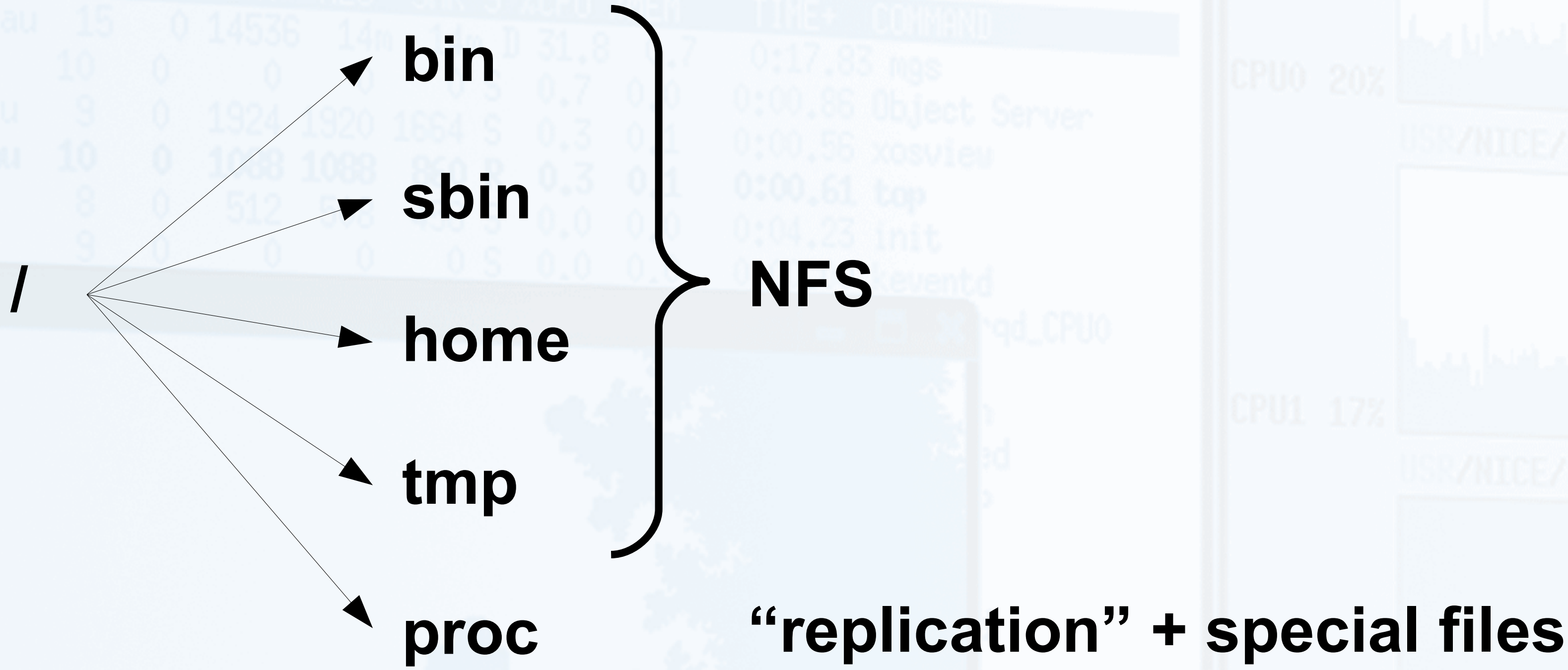
home NFS

tmp KerFS

proc “replication” + special files

# What is recommended currently

- “Simple” but that simply works !



# NFS Root: nice but not enough

- Using only NFS root
  - We have no choice to offer
  - Performance could be an issue
  - Installing an NFS-root server is not obvious
- Would be nice to
  - At least offer other choices
  - Offer a really integrated FS for Kerrighed

# Offering other choices

- We have plenty of file system to test
  - Free systems
    - GFS
    - OCFS
    - NFS V4
      - NFS V4.1 for parallel NFS
    - Lustre (probably too complex)
    - CIFS !
    - Etc...
  - Proprietary systems
    - SFS (SeaNodes FS + Hexanode)
    - Isilon solutions
    - Etc...
- Quite a lot of work to check all these FS

# Issue #1 : Existing DFS are not Kerrighed aware

- Cluster wide coherence of file data
  - What happens if 2 processes on 2 different nodes access the same file at the same time ?
    - NFS : data is not coherent during a short period of time
    - GFS : data is coherent, but the coherence is expensive
- File access performance
  - NFS : poor
  - GFS : good if good file access locality
- Unix sockets does not work cluster wide !
  - This is not a file system problem
    - Solved with the migrable streams

# Offering an integrated file system

- That was the plan for KerFS
- Now we have the kDFS file system under implementation
  - Storage of data on different cluster nodes
  - Implementation “Kerrighed aware”
    - Use of KDDM
    - No bad effect with process migration
    - Possible to implement intelligent policies for data / process placement

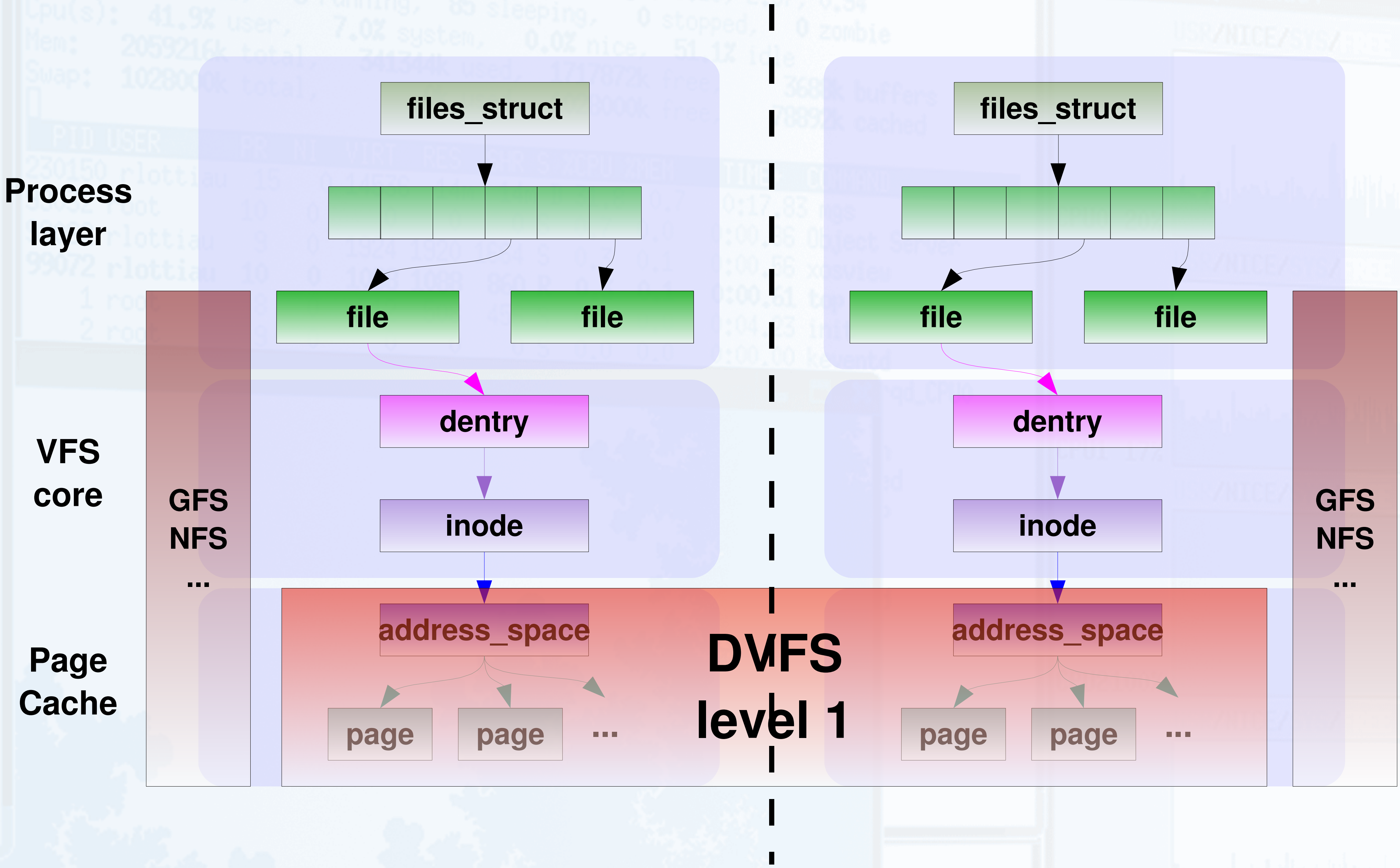
# Issue #2 : we don't want Yet Another Research FS

- Designing a production quality new FS from scratch is a very hard task
  - A whole company could be needed to achieve this
- The big question is
  - Would it be possible to reach a production quality FS ?
  - Yet another research FS is fun !
    - But it's just another fun research FS...
- If we consider node failures, the problem is even more complicated

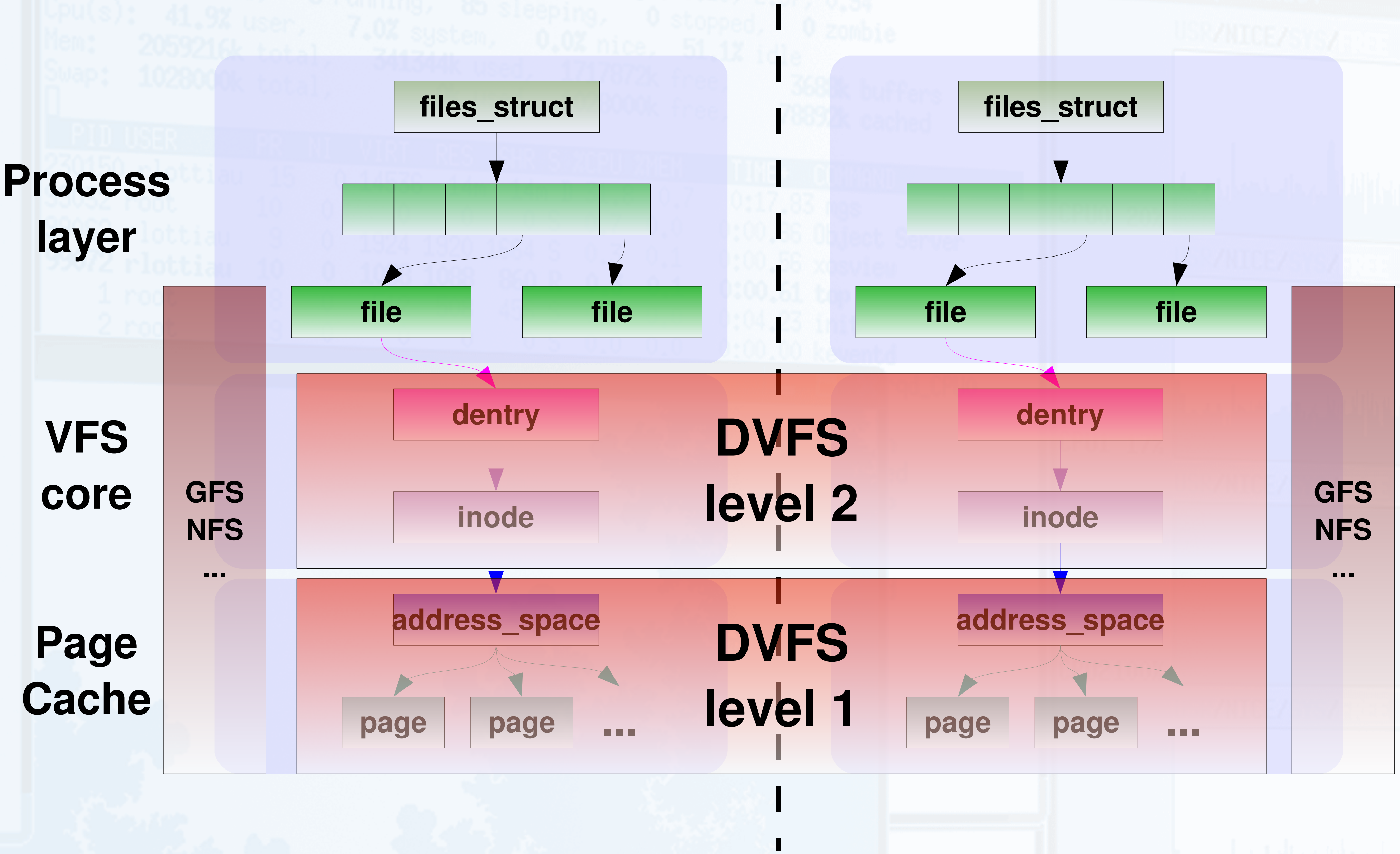
# A third approach: DVFS

- The main problem of existing DFS : data coherence
- Why there is no coherence problem without a DFS ?
  - No distributed copies of data : one unique file cache
- Solution : 1 unique cluster wide file cache !
  - Kerrighed DVFS level 1
- Data has no more to be written back to the server or the shared disk to be accessed remotely
- Data can be accessed directly from one node memory to another node memory

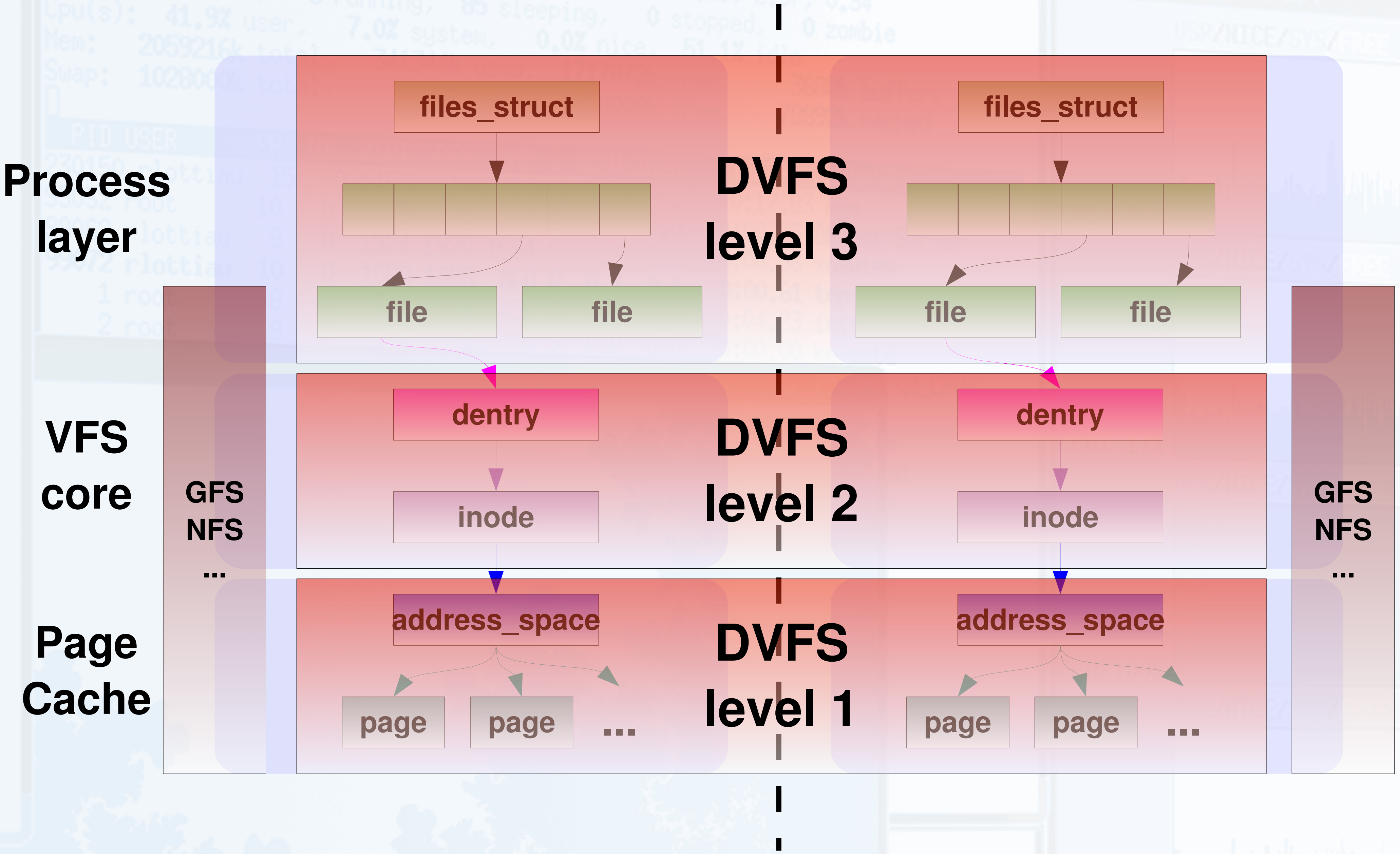
# Linux VFS – Kerrighed DVFS level 1



# Linux VFS – Kerrighed DVFS level 2



# Linux VFS – Kerrighed DVFS level 3



# DVFS: Nearly perfect solution ?

- Kerrighed aware architecture
- Can be plugged to “any” existing FS
- New DFS can be implemented within this framework
- Simpler to implement than a all new brand DFS
- This is not a new FS
  - This is more acceptable for customers

# Issue #3: DVFS

- Nothing done yet
- No manpower to do that
- Simpler than a DFS but quite complex anymay