Cost-Effective Virtual Petabytes Storage Pools

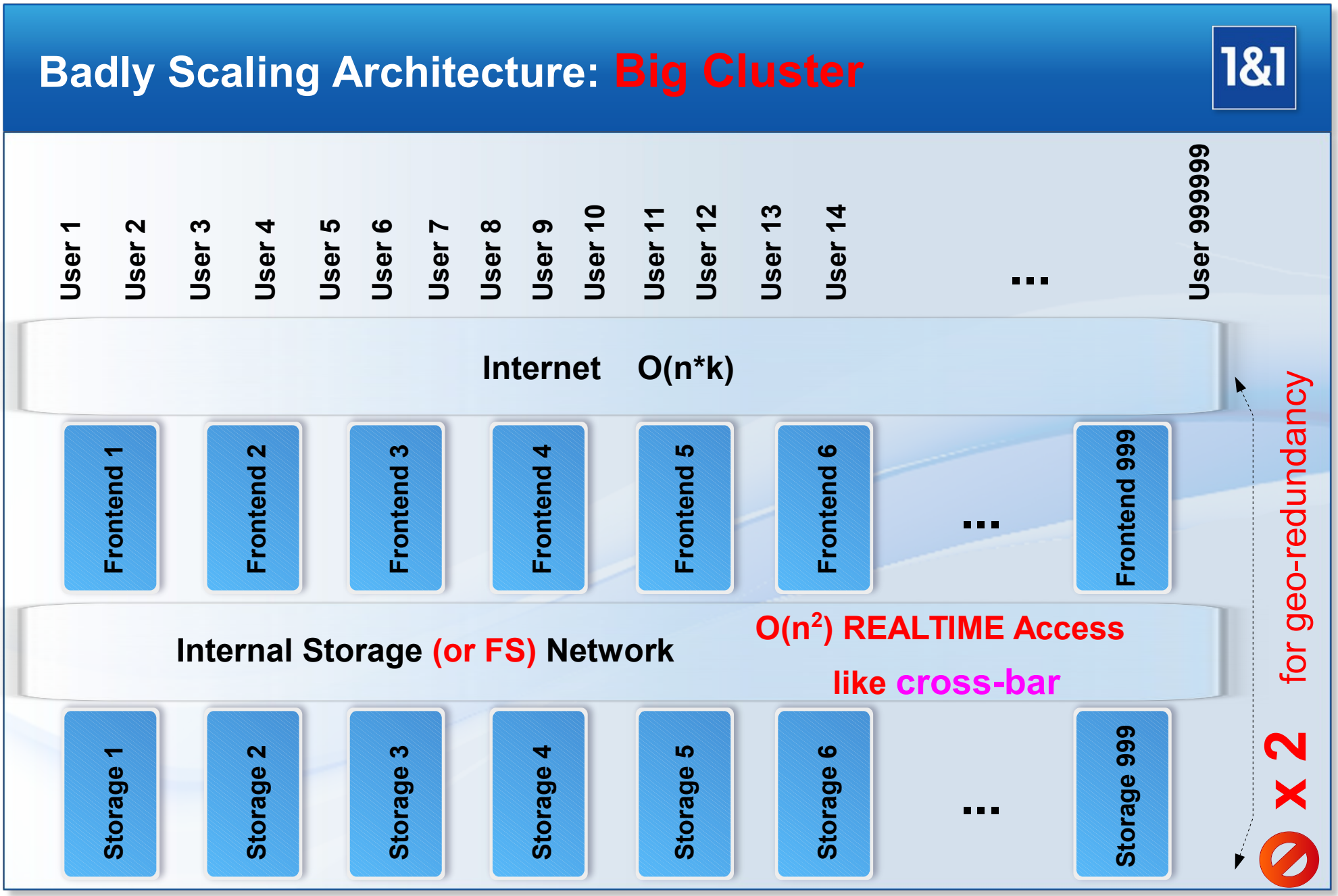
using MARS

FrOSCon 2017 Presentation by Thomas Schöbel-Theuer

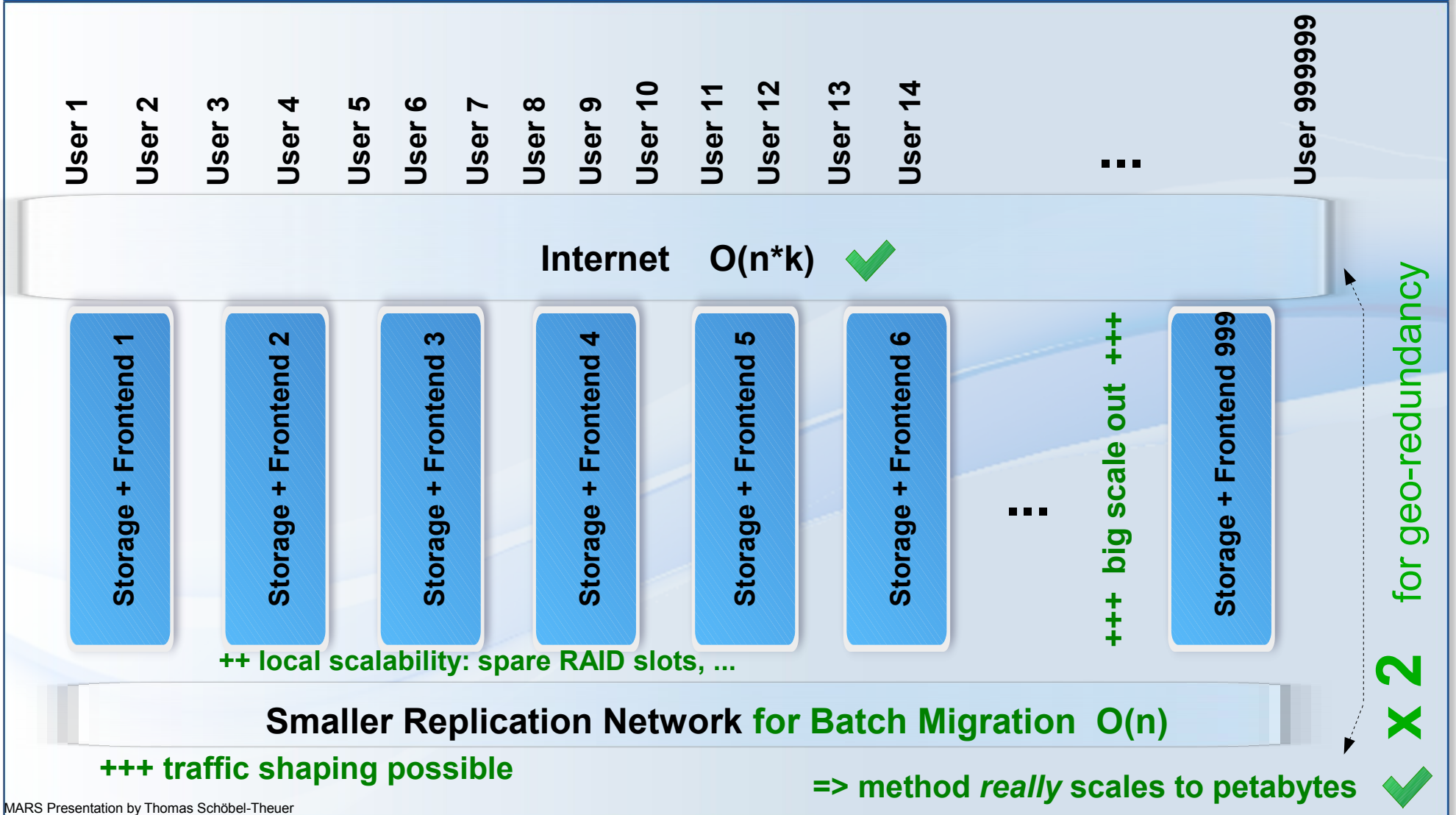
Virtual Petabytes Storage Pools: Agenda

- Scaling Properties of Storage Architectures
- Reliability of Storage Architectures
- Motivation: *Costs*
- Flexible MARS Sharding + Cluster-on-Demand
- Load Balancing by Background Data Migration
- Current Status / Future Plans

Badly Scaling Architecture: Big Cluster



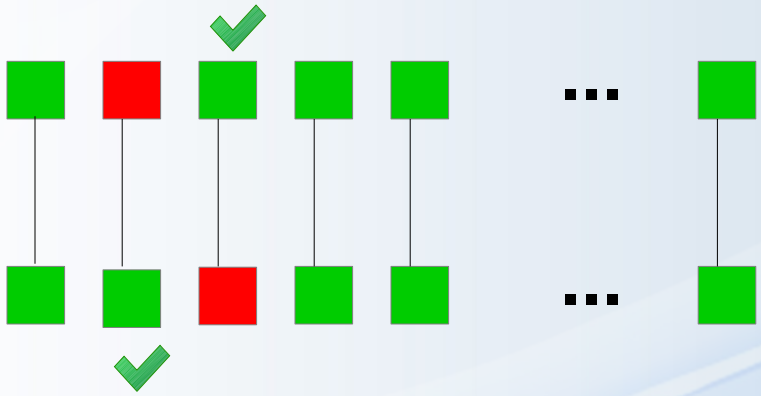
Well-Scaling Architecture: Sharding



Reliability of Architectures: NODE failures

2 Node failure => ALL their disks are unreachable

DRBD or MARS simple pairs

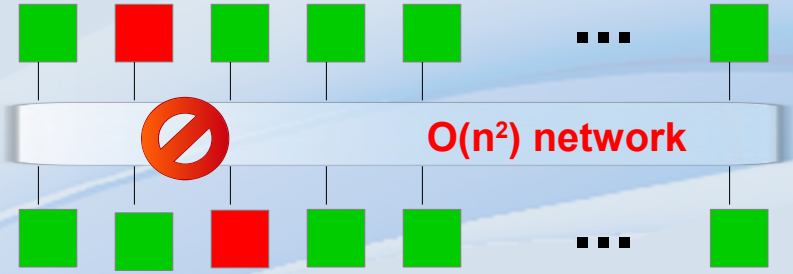


=> no customer-visible incident

Low probability for hitting the *same* pair, even then: only 1 shard affected => low total downtime

same n

Big Storage Cluster e.g. Ceph, Swift, ...



k=2 replicas not enough => INCIDENT because objects are randomly distributed across whole cluster

Higher probability for hitting *any* 2 nodes, then O(n) clients affected => much higher total downtime

need k >= 3 replicas here

Costs (1) non-georedundant, n>100 nodes

- Big Cluster:
  Typically ≈RAID-10 with k=3 replicas for failure compensation
- Disks: > 300%
- Additional CPU and RAM for storage nodes
- Additional power
- Additional HU

- Simple Sharding:
  Often local RAID-6 sufficient (plus external backup, no further redundancy)
- Disks: < 120%
- Client == Server
  no storage network
  MARS for LV background migration
- Hardware RAID controllers with BBU cache on 1 card
- Less power, less HU

Costs (2) georedundant => LONG Distances

- Big Cluster:
  - 2X ≈ RAID-10 for failure compensation
    (k=6 replicas needed, smaller does not work in long-lasting DC failure scenarios)
- Disks: > 600%
- Additional CPU and RAM for storage nodes
- Additional power
- Additional HU

- Geo-redundant Sharding:
  - 2 x local RAID-6
  - MARS for long distances
    or DRBD for room redundancy
- Disks: < 240%
- Hardware RAID controllers with BBU
- Less power
- Less HU

Costs (1+2): Geo-Redundancy Cheaper than Big Cluster

- Single Big Cluster:
  - ≈RAID-10 with k=3 replicas for failure compensation
- O(n) Clients
  + 3 • O(n) storage servers
  + $O(n^2)$ storage network
- Disks: > 300%
- Additional power
- Additional HU

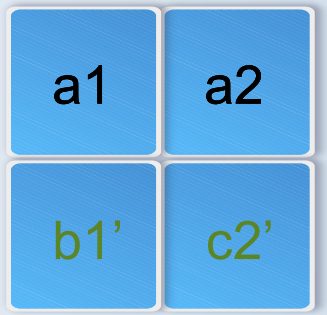
- Geo-redundant sharding:
  - 2 x local RAID-6
  - MARS for long distances

    or DRBD for room redundancy
- 2 • O(n) clients = storage servers
  + O(n) replication network
- Disks: < 240%
- Less total power
- Less total HU

  +++ geo failure scenarios

Costs (3): Geo-Redundancy even Cheaper

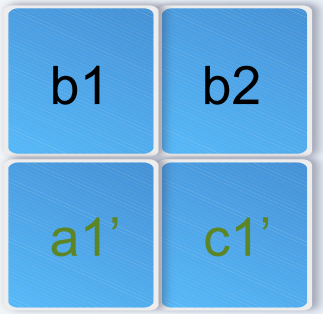
Precondition:
CPU must not be the bottleneck



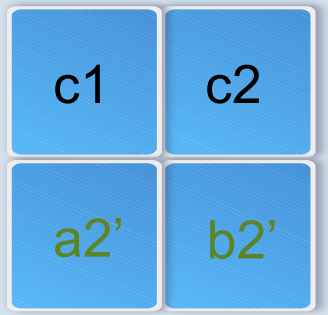
Datacenter 1

Idea: passive LV roles get less CPU

1 datacenter out of 3 may fail



Datacenter 2

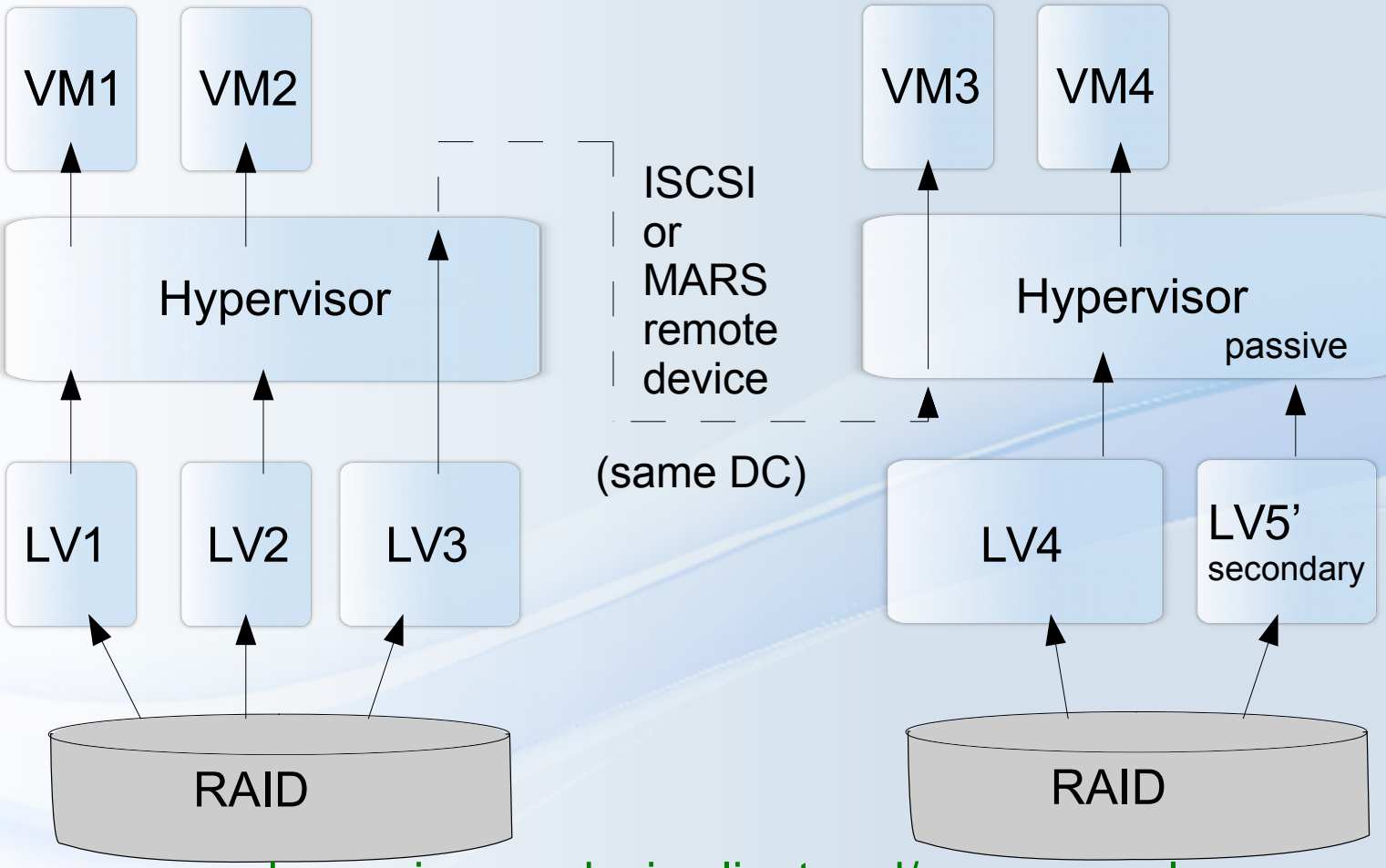


Datacenter 3

Total Storage: x 2
Total CPU: x 1.5
=> 1.5 • O(n)

HOWTO flexible CPU assignment => next slide

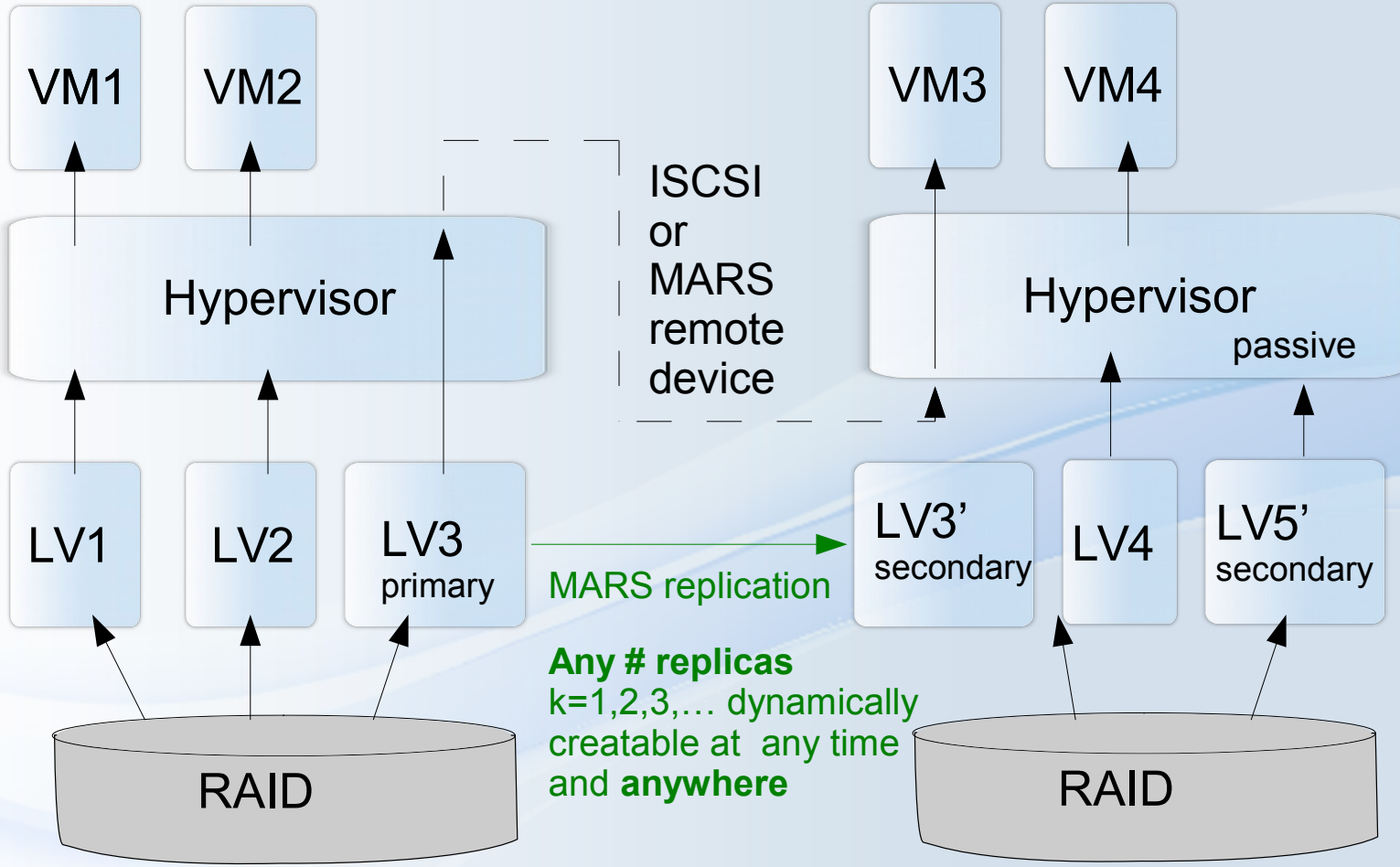
Flexible MARS Sharding + Cluster-on-Demand



any hypervisor works in client and/or server role

and preferably **locally** at the same time

Flexible MARS Background Migration



=> any hypervisor may be source or destination of some LV replicas at the same time

MARS Current Status

- MARS source under GPL + docs:

  `github.com/schoebel/mars`
  `mars-manual.pdf` ~ 100 pages

- mars0.1stable productive on customer data since 02/2014
- Backbone of the 1&1 geo-redundancy feature
- MARS status August 2017:

  > 2000 servers (shared hosting + databases)

  > 2x8 petabyte total

  ~ 10 billions of inodes in > 3000 xfs instances

  > 30 millions of operating hours

- New internal Efficiency project
  - Concentrate more LXC containers on 1 hypervisor
  - New public branch mars0.1b with many new features, e.g. mass-scale clustering, socket bundling, remote device, etc
  - mars0.1b currently in ALPHA stage

MARS Future Plans

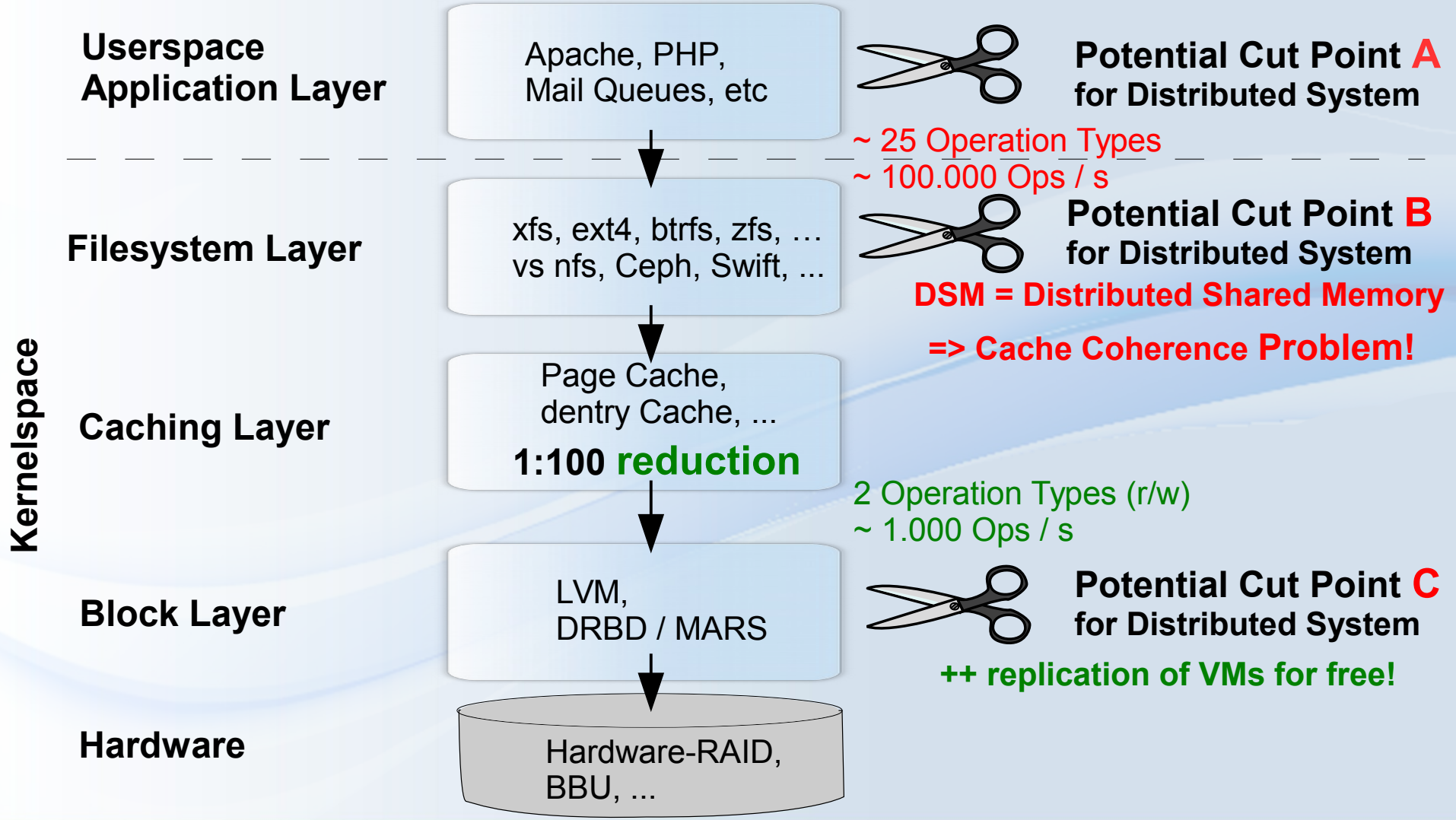
Automatic load balancing	TBD Separate implementation or libvirt / Openstack / Kubernetes plugins … ?
Virtual LVM-like Storage + VM pools	WIP 1&1 clustermanager `cm3` and/or libvirt plugin … ?
Physically sharded pools	Done MARS instead of DRBD

Collaboration sought

=> Opportunities for other OpenSource projects!

Appendix

Replication at Block Level vs FS Level



Use Cases DRBD+proxy vs MARS Light

DRBD+proxy
(proprietary)

Application area:

- Distances: any
- Aynchronously
  - **Buffering in RAM**
- Unreliable network leads to **frequent re-syncs**
  - RAM buffer gets lost
  - at cost of actuality
- **Long** inconsistencies during re-sync
- Under pressure: **permanent** inconsistency possible
- High memory overhead
- Difficult scaling to k>2 nodes

MARS Light
(GPL)

Application area:

- Distances: **any** (>>50 km)
- Asynchronously
  - near-synchronous modes in preparation
- Tolerates **unreliable network**
- Anytime consistency
  - no re-sync
- Under pressure: no inconsistency
  - possibly at cost of actuality
- Needs >= 100GB in `/mars/` for transaction logfiles
  - dedicated spindle(s) recommended
  - RAID with BBU recommended
- Easy scaling to k>2 nodes