R&D projects in CERN IT Storage Group

Dirk Duellmann, IT-DSS

CMS R&D meeting

12.Oct 15

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

R&D - a few non-technical comments

- Today most of IT department and DSS group focusses on ***service*** deployment
  - "D" takes place in small s/w development or dev-ops teams
    - separating dev from ops is often not a good idea
    - writing stable multi-threaded distributed services is not easy
  - "R" is hard to fund
    - focused on main technology risk factors for existing services
    - if possible in collaboration with
      - users - to (re-)define real service needs (eg EOS after WLCG jamborees)
      - technology providers - but clearly separated from any purchasing (eg CERN openlab)
      - universities - which share the academic interest (eg PhD program)
      - other sites - who share similar operational responsibilities
- Rest of this talk
  - a few concrete technical examples

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CERN Disk Storage Overview

	AFS	CASTOR	EOS	Ceph	NFS	CERNBox
Raw Capacity	3 PB	20 PB	140 PB	4 PB	200 TB	1.1 PB
Data Stored	390 TB	86 PB (tape)	27 PB	170 TB	36 TB	35 TB
Files Stored	2.7 B	300 M	284 M	77 M (obj)	120 M	14 M

AFS is CERN's linux home directory service

CASTOR & EOS are mainly used for the physics use case (Data Analysis and DAQ)

Ceph is our storage backend for images and volumes in OpenStack

NFS is mainly used by engineering application

CERNBox is our file synchronisation service based on OwnCloud+EOS

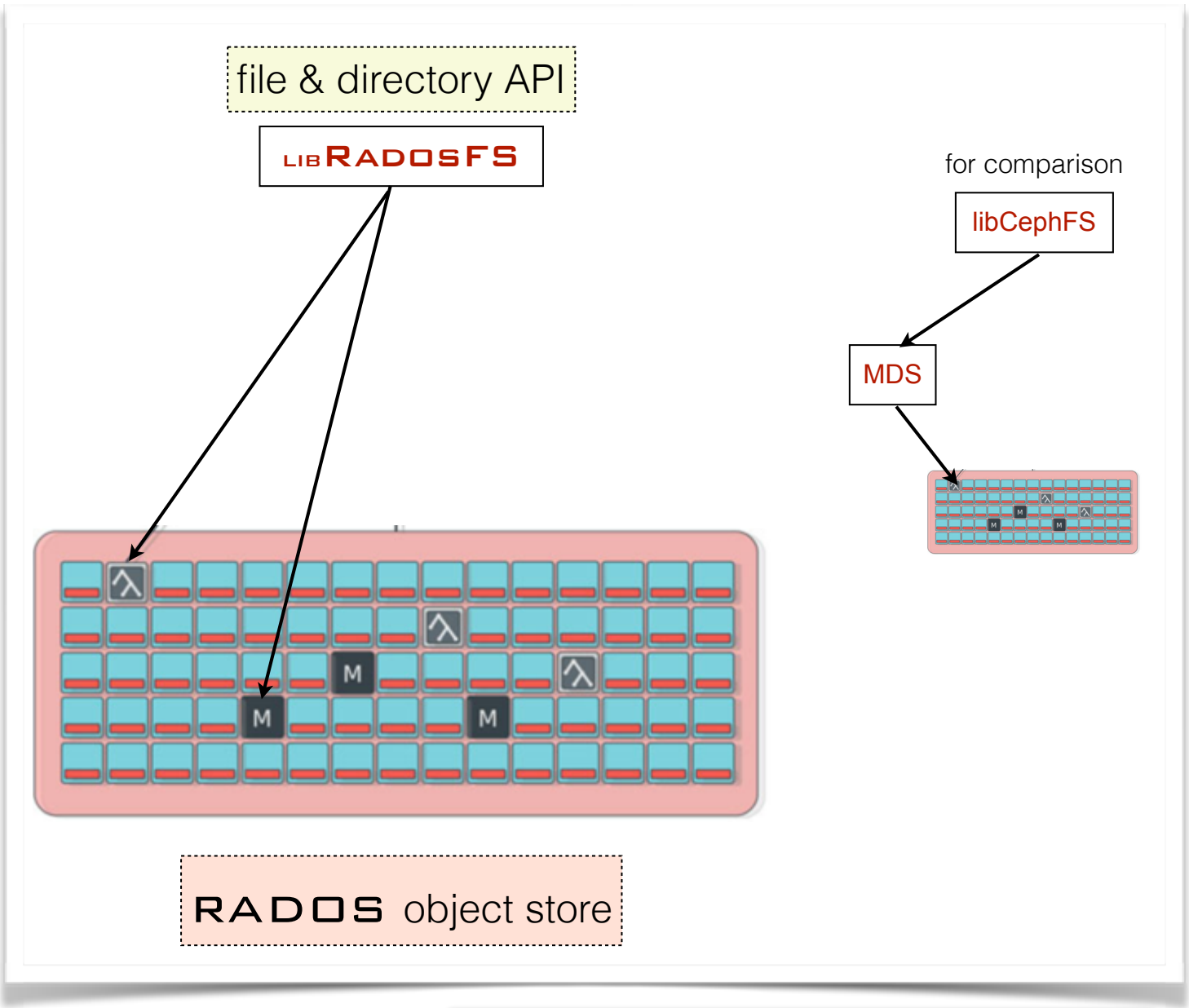
EOS Challenges

- EOS started with in-memory namespace
  - scalability
  - currently some 300 M files
    - new use cases like cernbox come with many small files
    - new usage patterns like RUCIO may introduce different balance between files and directories
  - latency of namespace restart may affect availability
    - tested up to 500 M files but cold-restart would be 20-30 mins
- Namespace R&D
  - change concept to scale out namespace?
    - eg via an distributed key-value / object store
    - {eg via federating different backend namespaces}
  - evaluate new storage media
    - eg using non-volatile memory (NV-RAM)

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

DSS Inline Meta-Data - libRadosFs

CERN IT Department



CERN fellow: Joaquim Rocha

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

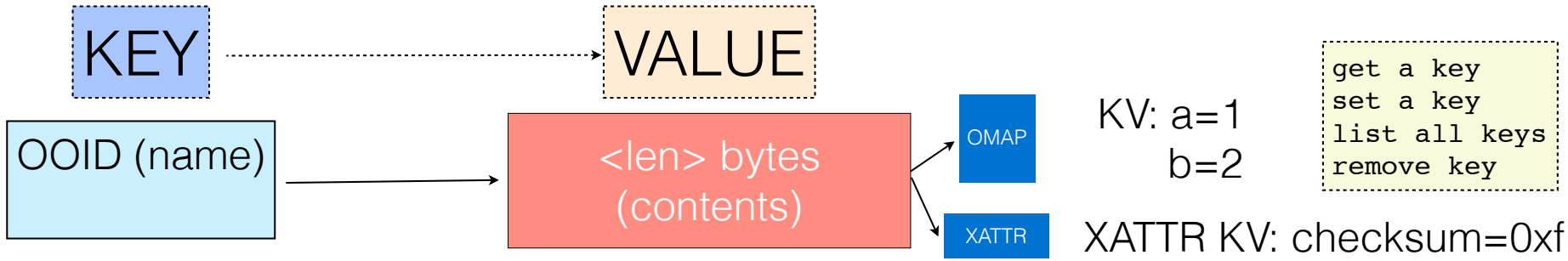
libRADOSFS

http://github.com/cern-eos/radosfs

- simple & lightweight C++ storage library
  - provides an **API** to store **files and directories** as objects in RADOS pools [Ceph]
  - using inodes for efficient renaming
  - no additional meta-data server
  - synchronous & asynchronous file IO & vector reads
  - file chunking - not striping - erasure coded pool support
  - small file inlining into directory objects
  - directory objects as WAL with auto-compaction
  - extended attribute support on files, directories and entries inside a directory
  - parallel query interface
  - store & commit - possibility to use file inodes and register them later into directories
  - fsck tool - check & optional repair
  - ...

RADOS

Ceph object storage API



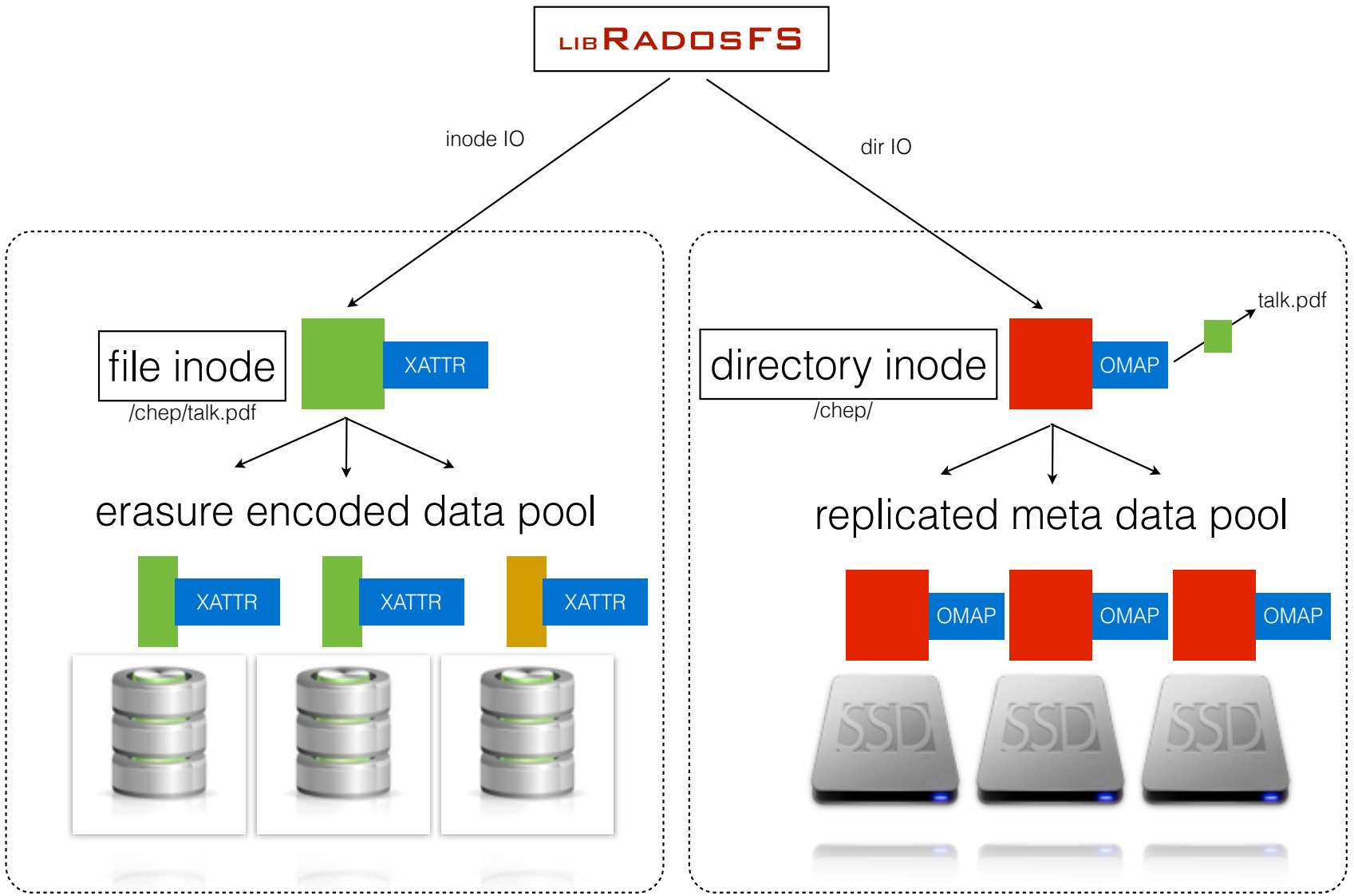
operations

```
read(offset,len)
write(offset,len)
writefull(len)
append(len)
remove()
locks
conditions
single object transactions
```

- each object provides
  - contents (value)
  - kev-value map (omap)
  - extended attribute map (xattr)
- erasure encoded objects support only
  - xattr map
  - full object writes or appends with the EC blocksize

LIBRADOSFS

file and directory IO

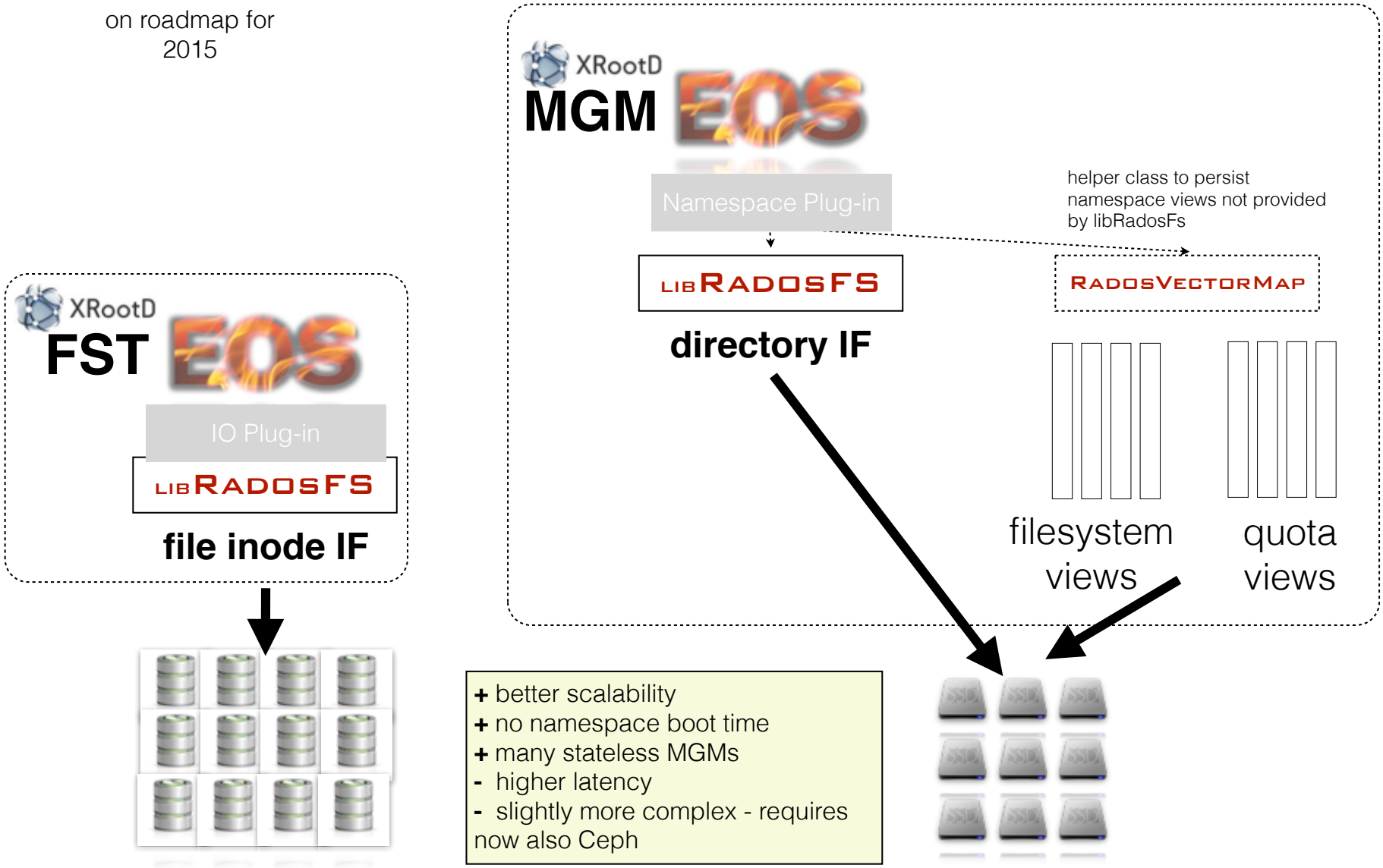


object contents: erasure encoded
object meta-data: replicated

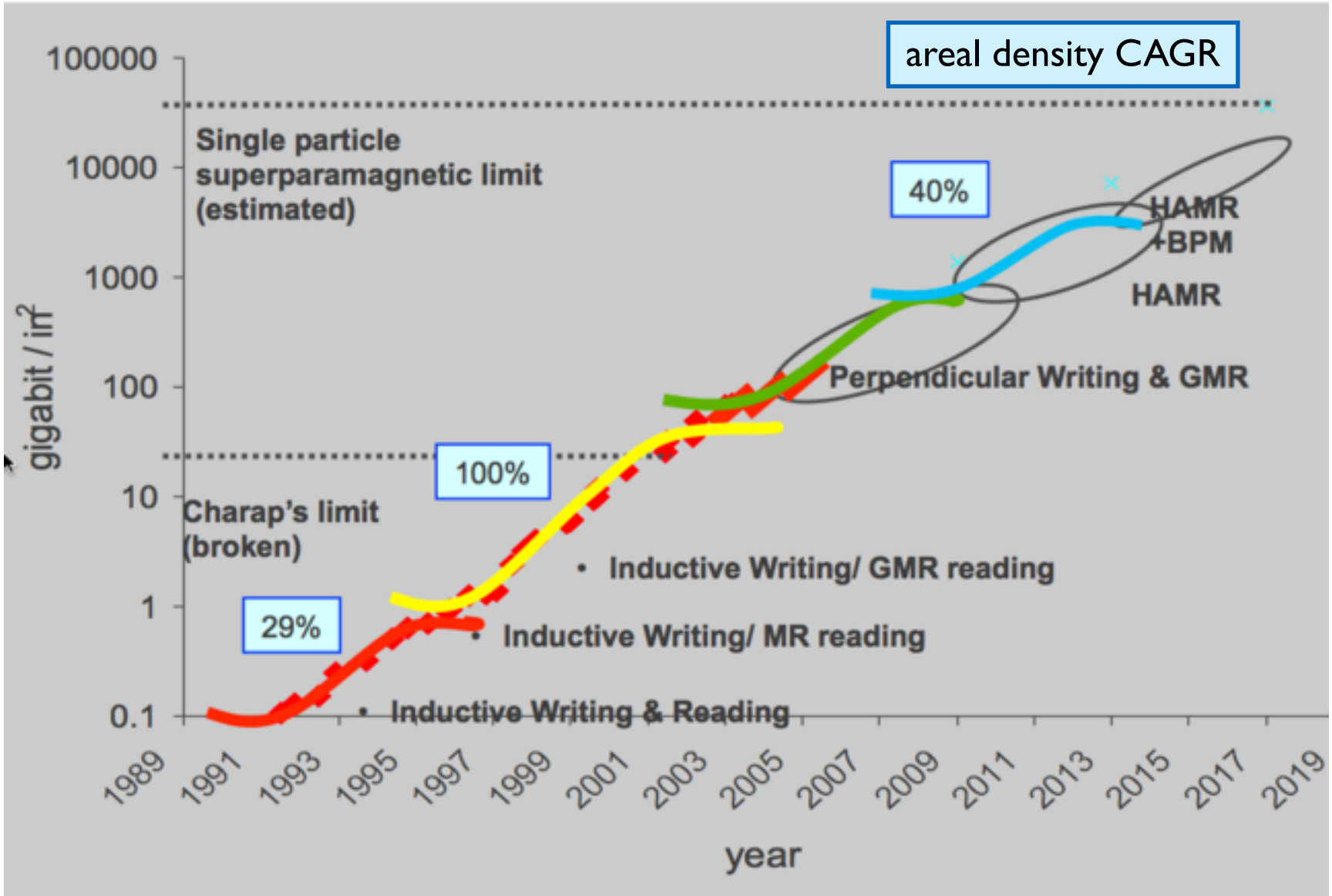
LIBRADOSFS

integration into EOS

on roadmap for 2015



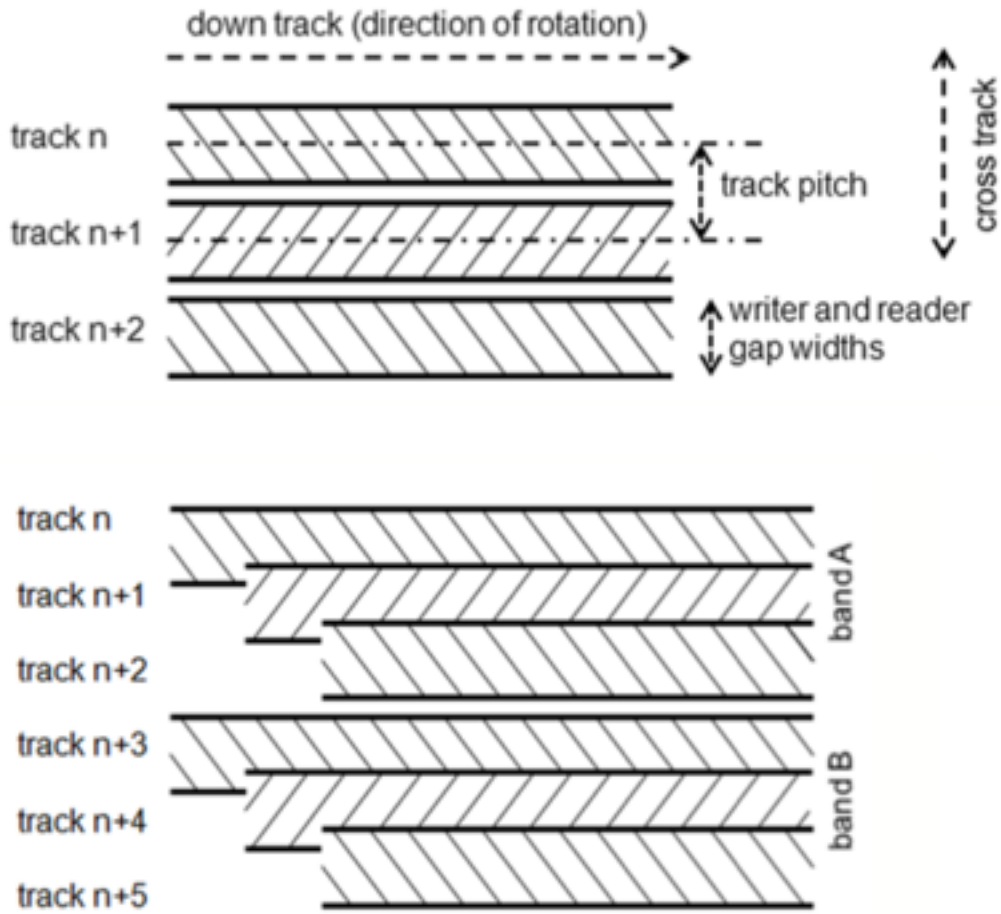
Does Kryder's law still hold?



source: HDD Opportunities & Challenges, Now to 2020, Dave Anderson, Seagate

Shingled Recording

- Shingled Media
  - wide write head
  - narrow read head
- Result
  - continued density increase
  - but, write amplification within a band



Impact of Shingled Recording

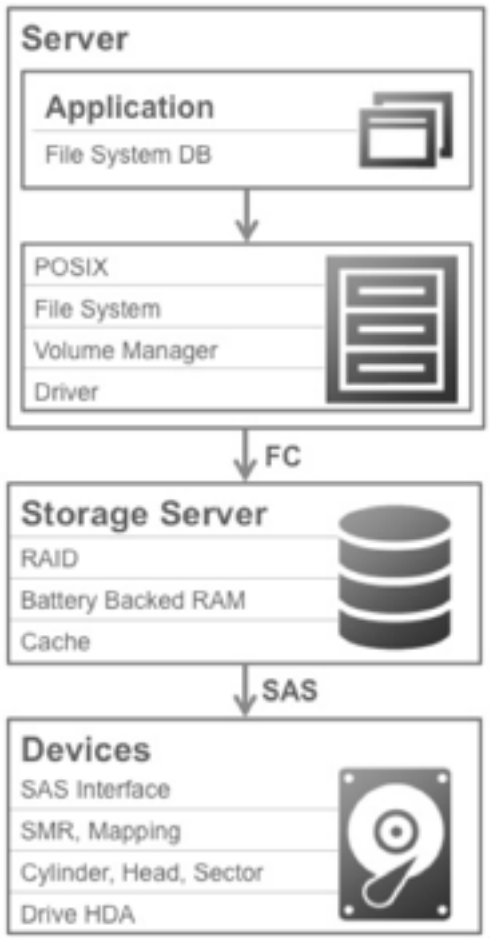
- Gap between read and write performance increases
  - need to check eg if metadata mixing with data is still feasible
- Market / application impact
  - Will there be several types of disks?
    - emulation of a traditional disk (!)
    - explicit band management by application (?)
    - constraint semantics / object disk (?)
- Open questions:
  - which types will reach a market share & price that makes them attractive for science applications ?
  - can the constrained semantics be applied to HEP workflows?

Object Disk

- Each disk talks object storage protocol over TCP
  - replication/failover with other disks in a networked disk cluster
  - open access library for app development

– Why now?
  - shingled media comes with constrained (object) semantic: eg no updates

– Early stage with several open questions
  - port price for disk network vs price gain by reduced server/power cost?
  - standardisation of protocol/semantics to allow app development at low risk of vendor binding?



Resident Seagate Contributor: Paul Lensing

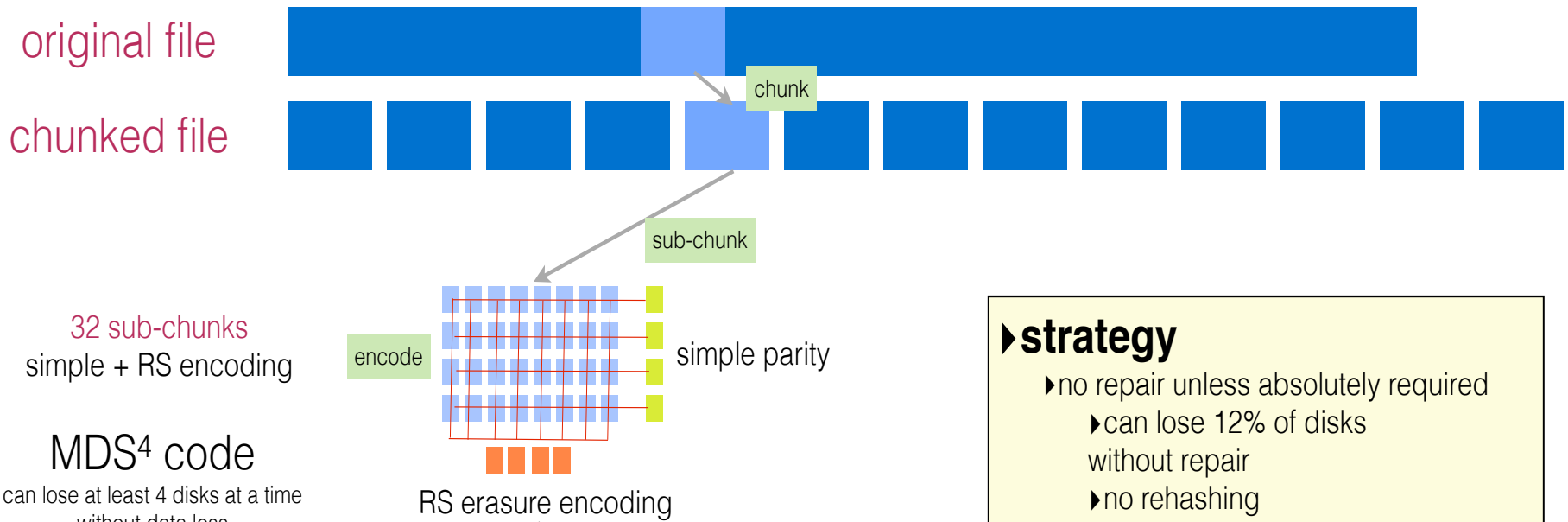
Seagate Kinetic API

- Kinetic API
  - Access Control
    - READ - can read
    - WRITE - can write
    - DELETE - can delete
    - RANGE - can do range
    - SETUP - can setup device
    - P2POP - can do p2p copy
    - GETLOG - can get log
    - SECURITY - can set security
  - NOOP - like ping
  - PUT - store object max. value size 1 MB
  - DELETE - delete object
  - FLUSH - flush outstanding PUT/DELETE to device (=sync)
  - GET - retrieve value + meta data
  - GETVERSION - retrieve version tag for object
  - GETNEXT - return next sorted key
  - GETPREVIOUS - return previous sorted key
  - GETKEYRANGE - return keys in range
  - SETCLUSTERVERSION - set cluetser version
  - SETPIN - instant secure erase
  - SECURITY - set ACL
  - GETLOG - retrieve log
  - PEERTOPEERPUSH - copy KV between drives

- API less feature rich than rados API - low-level
  - no partial value get/updates/append - only full object GET/PUT
  - no arbitrary map per object, but vector clock/version
  - no clustering support between devices, but P2P push
- protocol implemented with google protocol buffers
- disk uses sorted string tables and log structured merge tree technology

- need to implement high-level API & clustering software : **libkineticio**

File Encoding



32 sub-chunks
simple + RS encoding

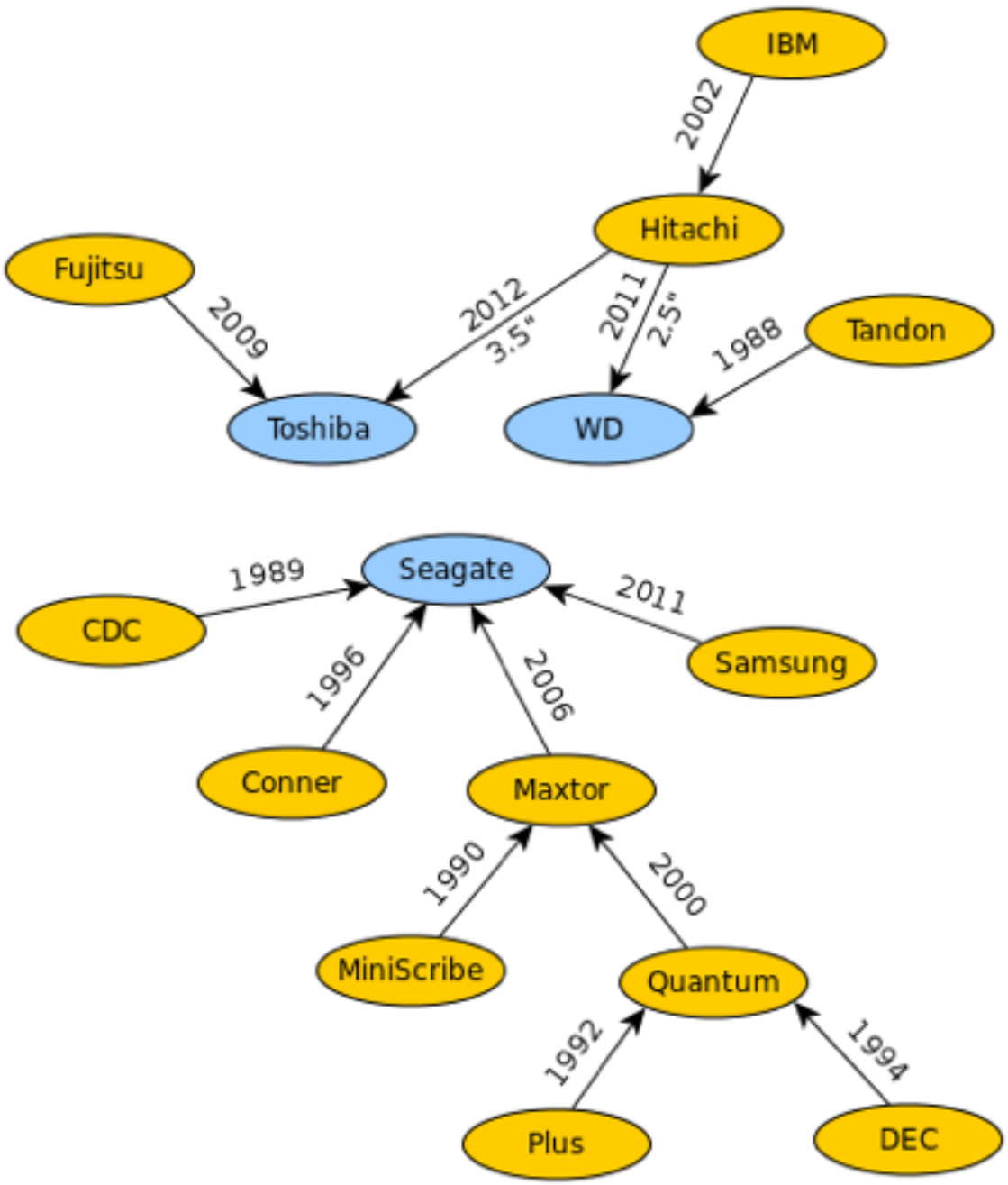
MDS[4] code
can lose at least 4 disks at a time without data loss

- **strategy**
  - no repair unless absolutely required
    - can lose 12% of disks without repair
    - no rehashing
      - only replace
      - overflow to next neighbor

(32,4,4) encoding has 25% volume overhead

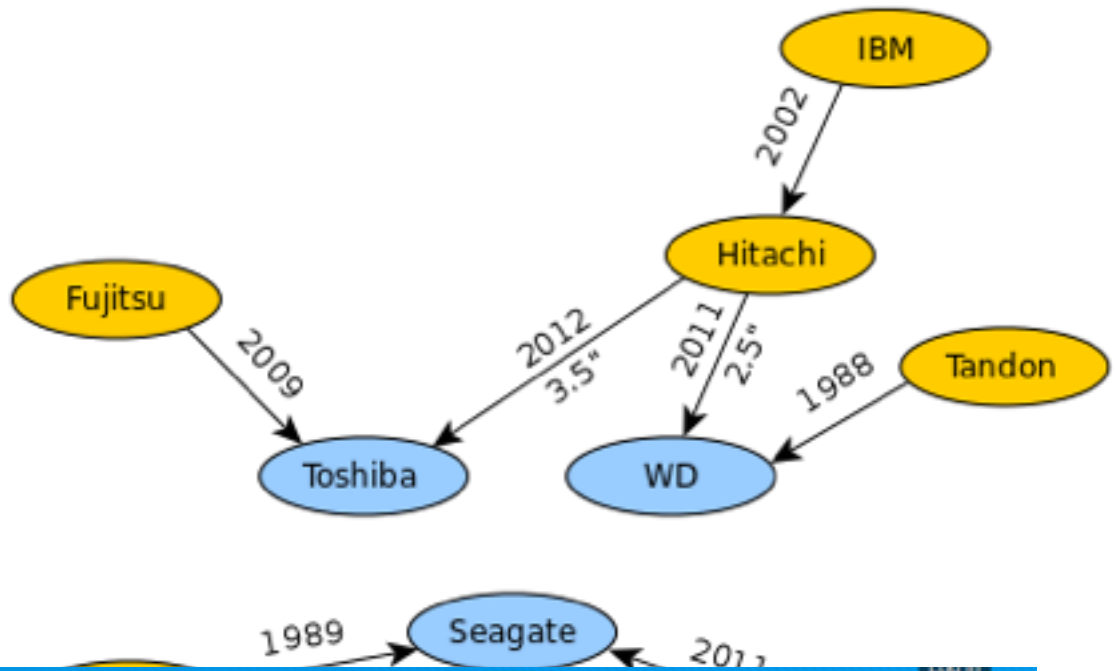
best kinetic performance for 32M chunks = 1M sub-chunks

Disk Market Consolidation



CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Disk Market Consolidation



LINUX FOUNDATION

Share 日本語 LF Sites

HOME | TRAINING | EVENTS | COLLABORATIVE PROJECTS

About Us Join News & Media Programs Workgroups Publications

Home › Linux Foundation Brings Together Industry Leaders to Advance Cloud Object Storage Technologies

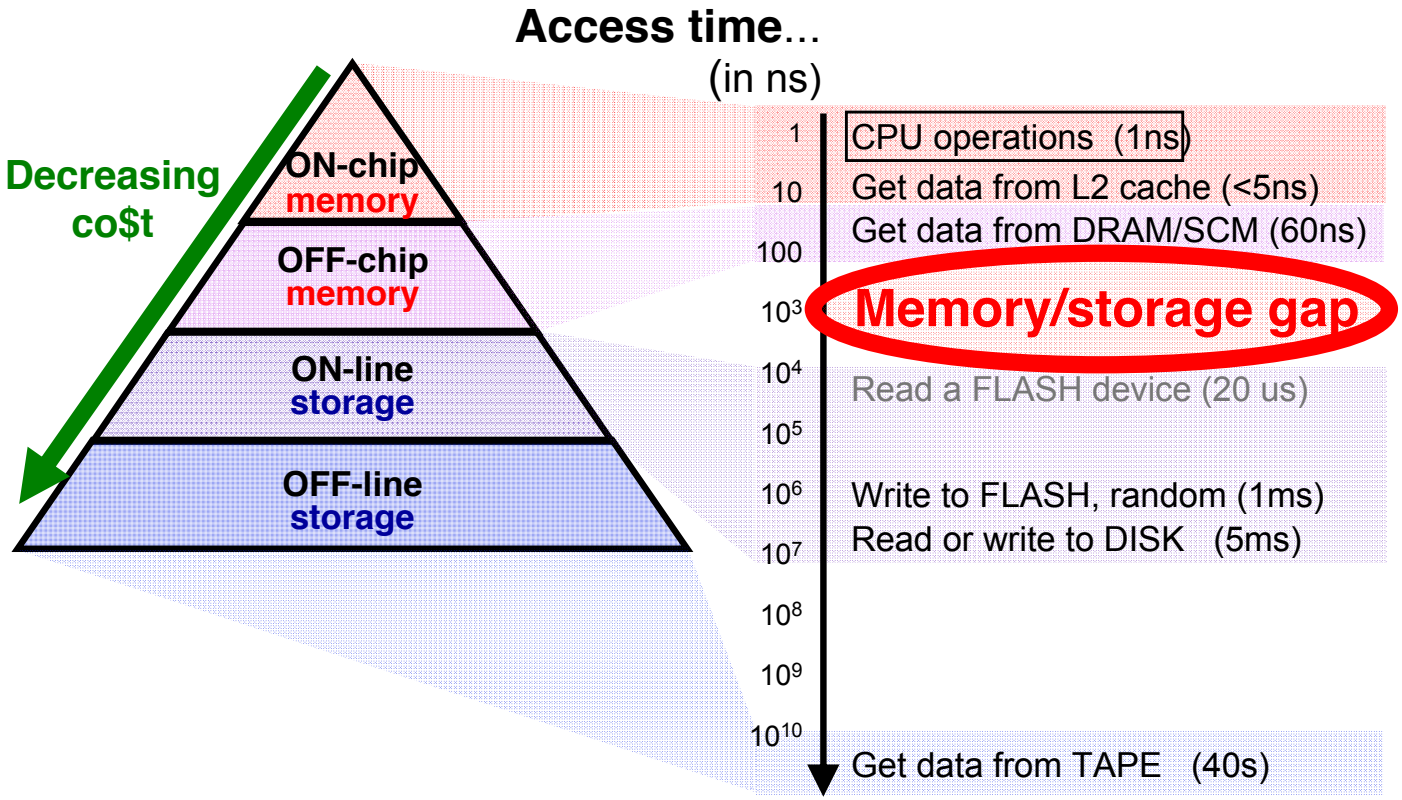
Email

Linux Foundation Brings Together Industry Leaders to Advance Cloud Object Storage Technologies

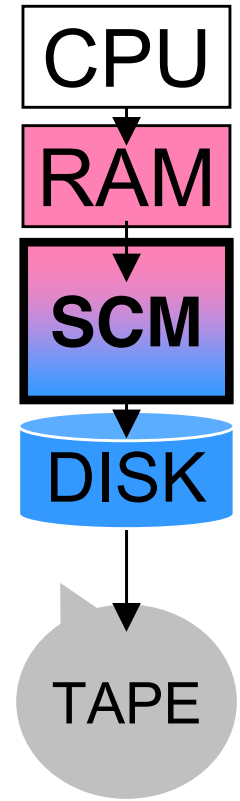
By Linux_Foundation - August 17, 2015 - 8:00am

Industry seeks to advance object-based, software-defined storage on Ethernet-enabled storage devices with support from Cisco, Cleversafe, Dell, Digital Sense, Huawei, NetApp, Open vStorage, Red Hat, Scality, Seagate, SwiftStack, Toshiba and Western Digital

Problem (& opportunity): The access-time gap between memory & storage



Near-future



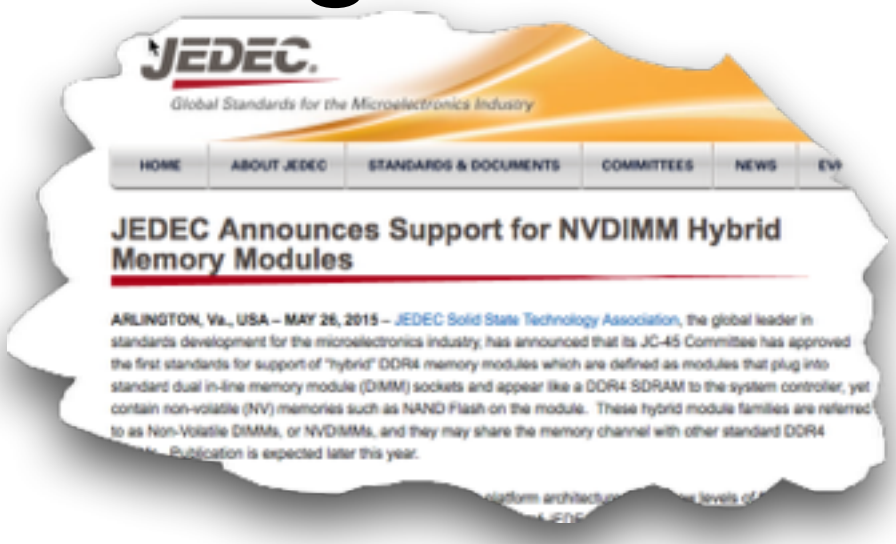
Research into new solid-state non-volatile memory candidates
– originally motivated by finding a "successor" for NAND Flash –
has opened up several interesting ways to change the memory/storage hierarchy…

1) **Embedded** Non-Volatile Memory – low-density, fast ON-chip NVM
2) **Embedded** Storage – low density, slower ON-chip storage

3) M-type **Storage Class Memory** – **high-density**, fast OFF- (or ON*)-chip NVM
4) S-type **Storage Class Memory** – **high-density**, very-near-ON-line storage

* ON-chip using 3-D packaging

R&D: non-volatile memory is coming! but how do we use it ?



- still early days for products, but software integration can already be prototyped
  - transactional memory
  - use an SSD-based filesystem
- CERN openLab project on NV-RAM based catalogue with Data Storage Institute Singapore

openLab Summer Student: Tobias Kappé

Moving the EOS namespace to persistent memory

Tobias Kappé (IT-DSS-DT)
tkappe@cern.ch

Supervised by
Elvin Alin Sindrilaru

Data Storage Institute

Mnemosyne

Mnemosyne[1] exposes persistency to C/C++:

- `pstatic` variables are stored persistently
- `pmalloc`/`pfree` allocate persistent memory
- `persistent` annotations ensure correctness
- `atomic` blocks allow transaction control

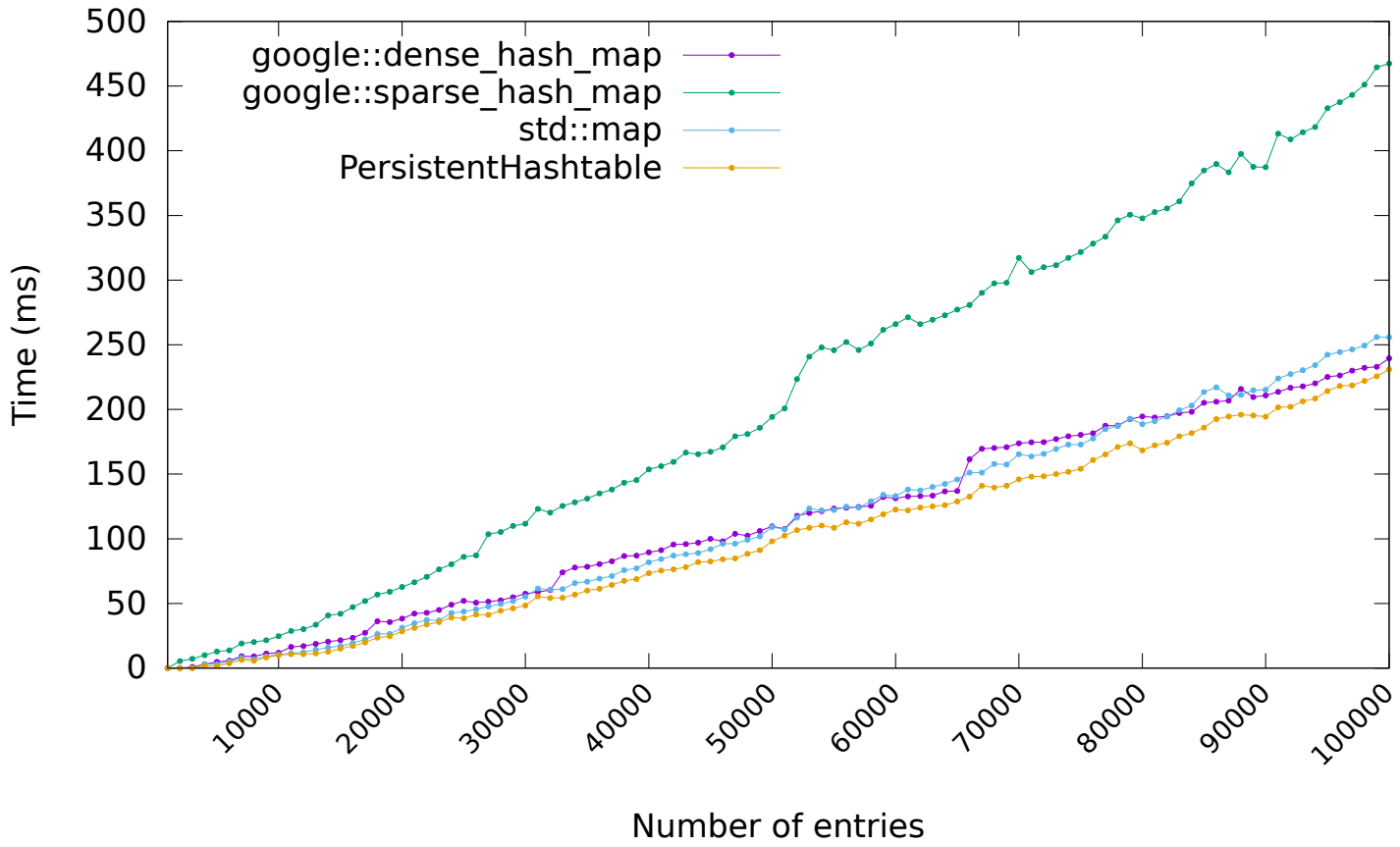
[1] Volos, Tack and Swift (2011)

Mnemosyne

Simplified example: (courtesy of Sergio Ruocco and Le Duy Khanh, DSI)

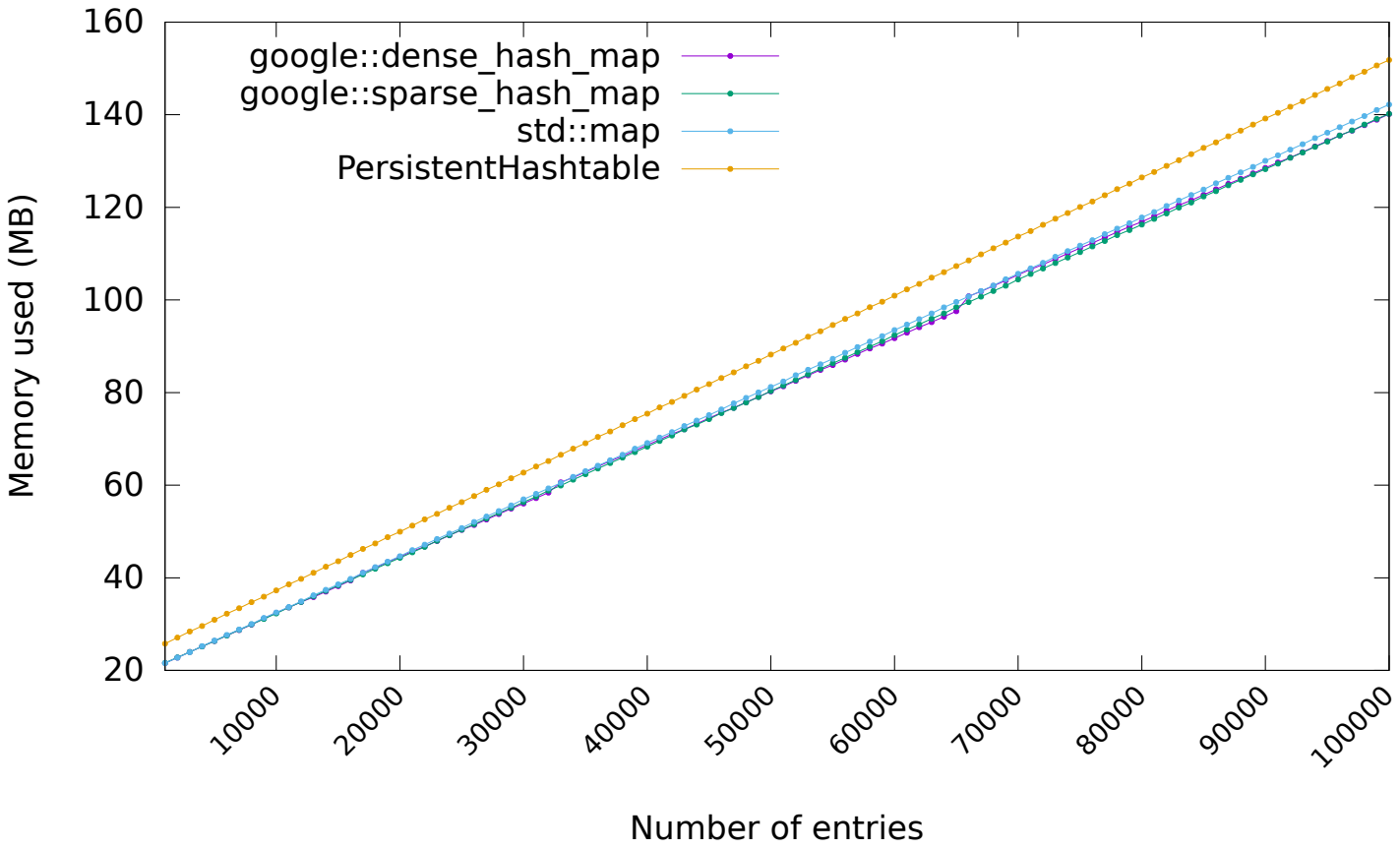
```
pstatic int persistent * p_ptr;

int main (int argc, char const *argv[]) {
    if (p_ptr == NULL) {
        atomic {
            p_ptr = pmalloc(sizeof(int));
            *p_ptr = 0;
        }
    } else {
        atomic { *p_ptr += 1; }
    }
    printf("*p_ptr = %d\n", *p_ptr );
    return 0;
}
```

Hashtable performance



PersistentHashtable scales and can match google::dense_hash_map!

Hashtable memory usage



PersistentHashtable has more memory overhead (due to the AVL tree).

How can we optimise our systems further?

- Infrastructure analytics
  - apply statistical analytics to complete system: storage, cpu, network, user app
  - measure quantitative impact of changes on real jobs
  - predict problems and outcome of planned changes
- Easy!
  - looks like physics analysis with infrastructure metrics instead of physics data
  - … really?

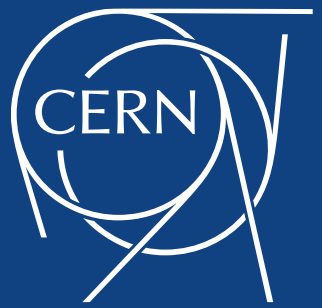
Non-trivial because…

- Technically
  - needs consolidated service and application side metrics
    - in production: Flume, HDFS, MR, Pig, HBase, Spark, …
    - data collection shared with operational monitoring (Elastic Search)
- Conceptually
  - some established metrics turn out to be less suitable for analysis of a large ensemble with different & varying workloads
    - "cpu efficiency" = t_cpu / t_wall ?
    - "storage efficiency" = GB/s ?
  - correlation in time does not imply causal relation
- Sociologically
  - better observe "rule of local discovery"
  - people who quantitatively understand infrastructure metrics are busy running services — Always …

Gentner/PhD student: Christian Nieke

Initial findings & surprises

- Some hidden / unknown (ab-)use patterns
  - really hot files: replicate file and access paths
  - really remote files: some users try working via the WAN (eg 120 ms RTT without enabling vector reads etc)
  - software bugs: users writing 1PB a day in two replicas without noticing
- In large distributed systems neither users nor service providers alone can easily spot even significant optimisation options
  - started expert working group across IT services and experiments



www.cern.ch

Thank you! Questions?