

Integrating scale-out storage & ethernet drive technology into

Paul Lensing

Joaquim Rocha

Dan van der Ster

Andreas-Joachim Peters

IT Data & Storage Service Group

CHEP Okinawa 13.-17.4.2015

XRootD

HTTPS
SRM
WebDAV
HTTP OwnCloud
FUSE gridFTP

EOS
SRM
HTTPS OwnCloud

EOS

Disk Storage @ CERN

CONTENTS



- new challenges for EOS
- R&D projects
 - **libradosfs**
 - a meta-data server free storage library
 - integration into EOS
 - **libkineticio**
 - software defined storage with ethernet disk drives
- personal vision of a new generation of CERN storage
- summary & outlook





CHALLENGES



- new use cases bring **many more small files** to EOS



- instances **not full** - we have 140 PB deployed - 38% used
 - expect 2.5x more files anyway when all disks full
 - means **~300 M files** in largest instance

- **more directories**

ATLAS Rucio naming convention multiplies number of directories

[worst case 1 file = 1 directory]

- namespace **scalability** limit

- meta data server tested to work with 0.5 B files, however service restart can take 20-30 minutes
- need to modify the implementation of the namespace boot
 - the namespace write-ahead log file is read on service start up and inserted in-memory into sets, vector and map structures
 - **can we persist directly sets, vector and maps?**
 - **can we create a stateless meta-data server for horizontal scaling?**

► implement namespace on top of scale-out storage : **libradosfs**

LIBRADOSFS

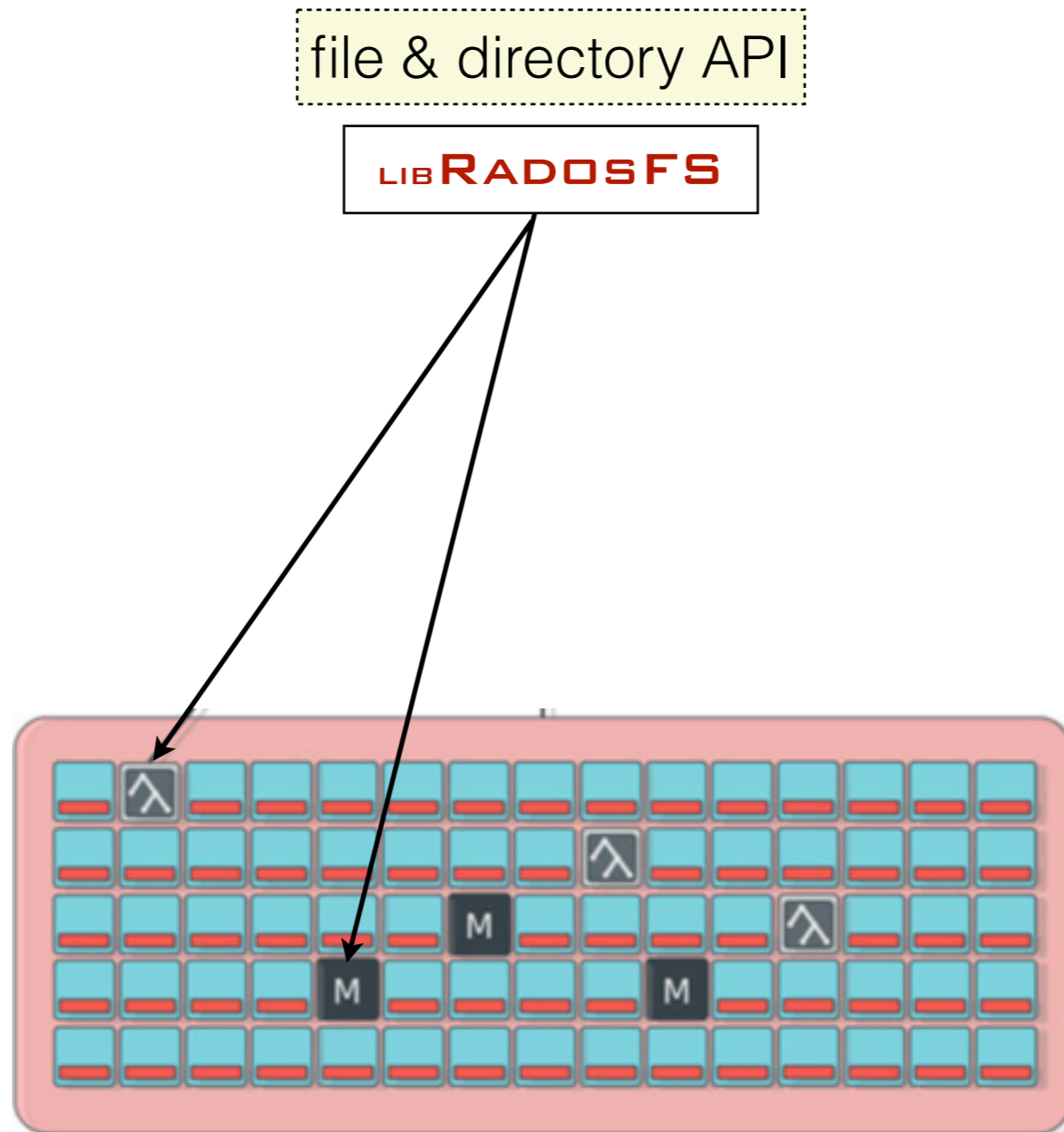
meta data server free storage library

LIBRADOSFS

a scalable filesystem library

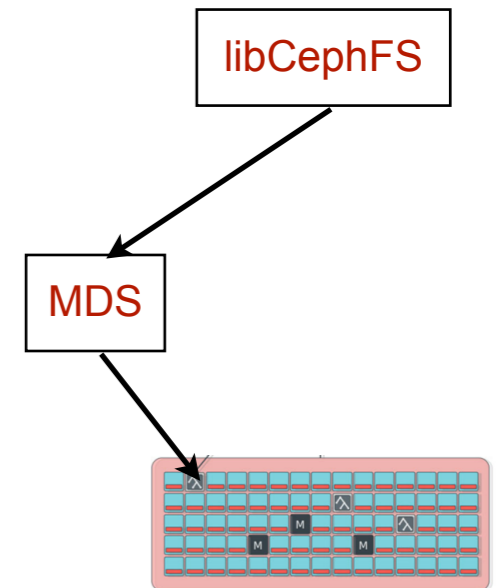


meta data server
free file-system



RADOS object store

for comparison



LIBRADOSFS

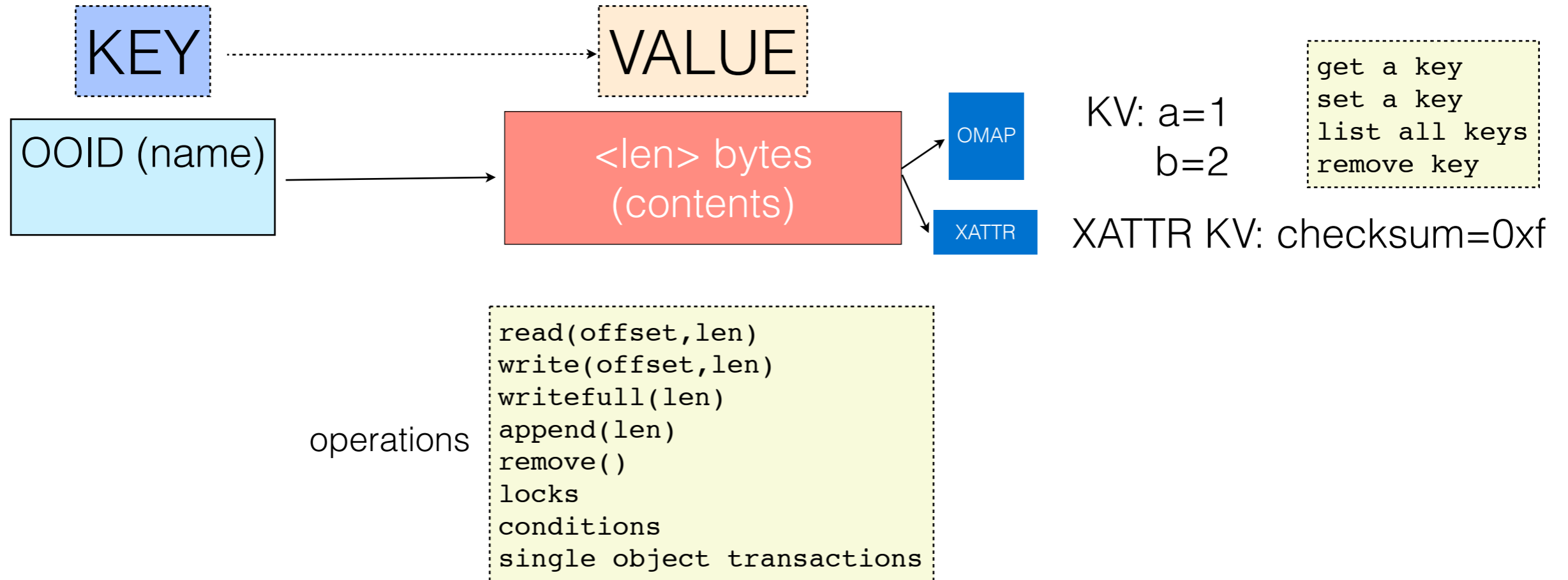


<http://github.com/cern-eos/radosfs>

- 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

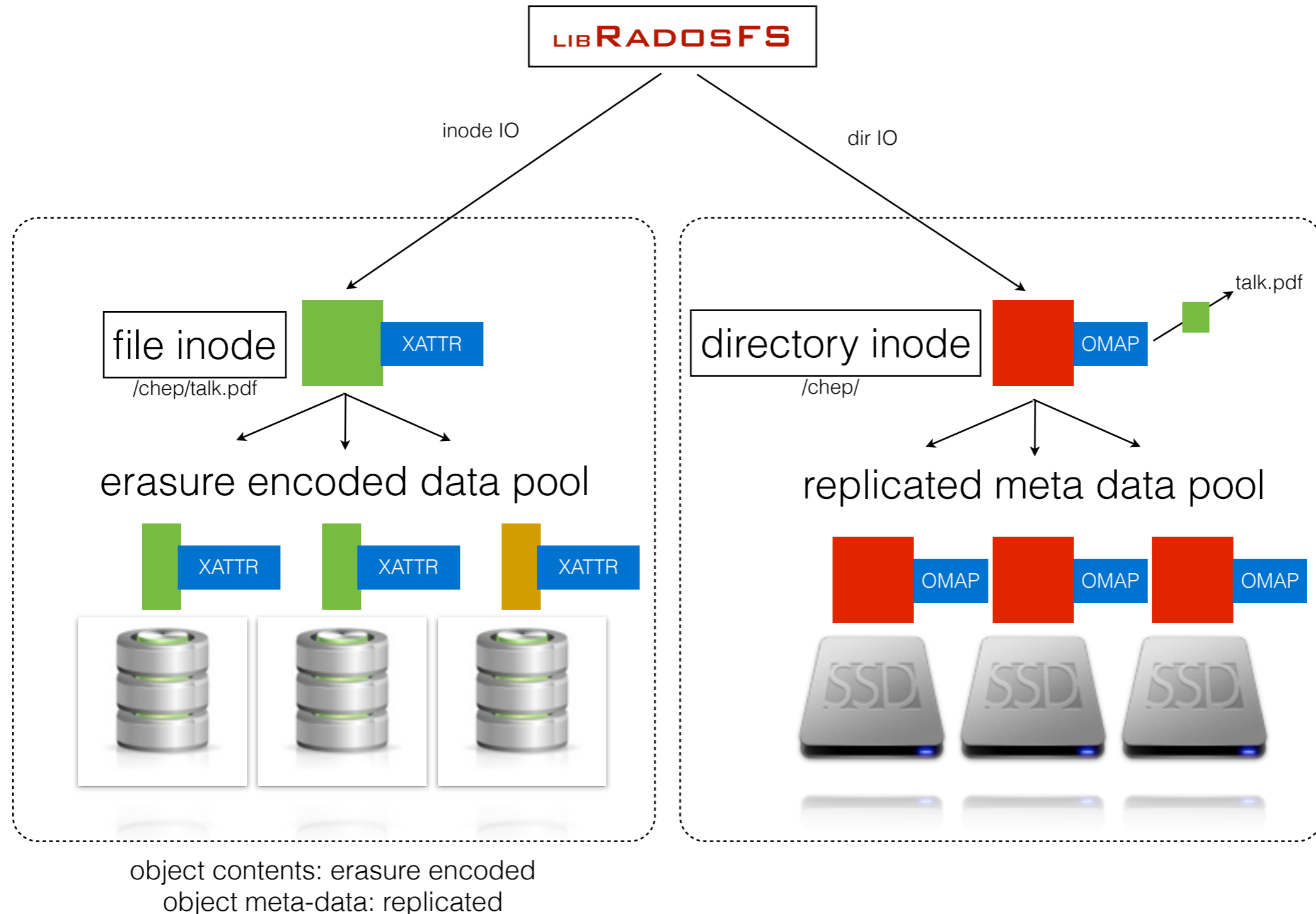


- each object provides
 - ▶ contents (value)
 - ▶ key-value map (omap)
 - ▶ extended attribute map (xattr)
- erasure encoded objects support only
 - ▶ xattr map
 - ▶ full object writes or appends with the EC blocksize

LIBRADOSFS



file and directory IO

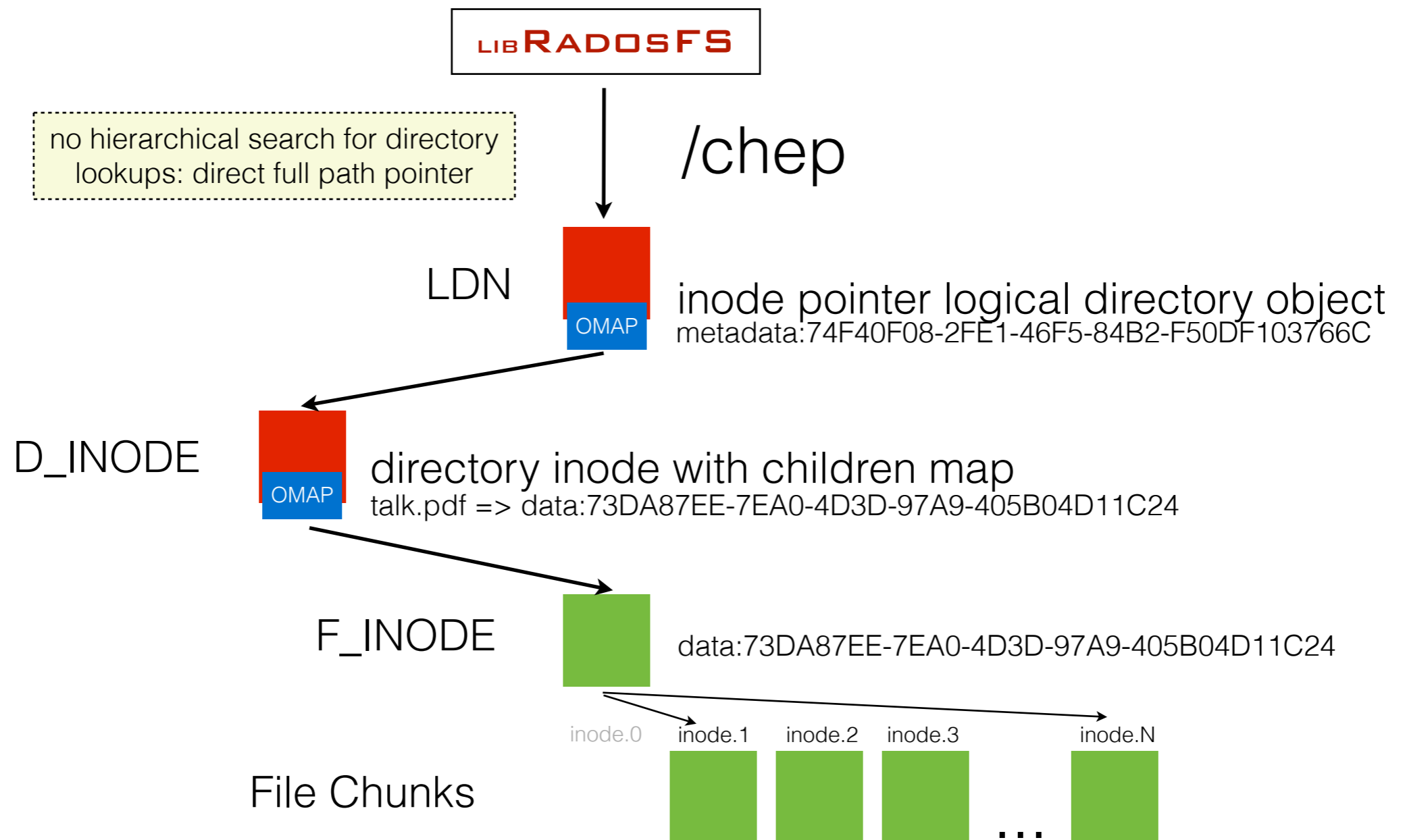


LIBRADOSFS

directory/file lookup



▶ How do I find the file /chep/talk.pdf ?



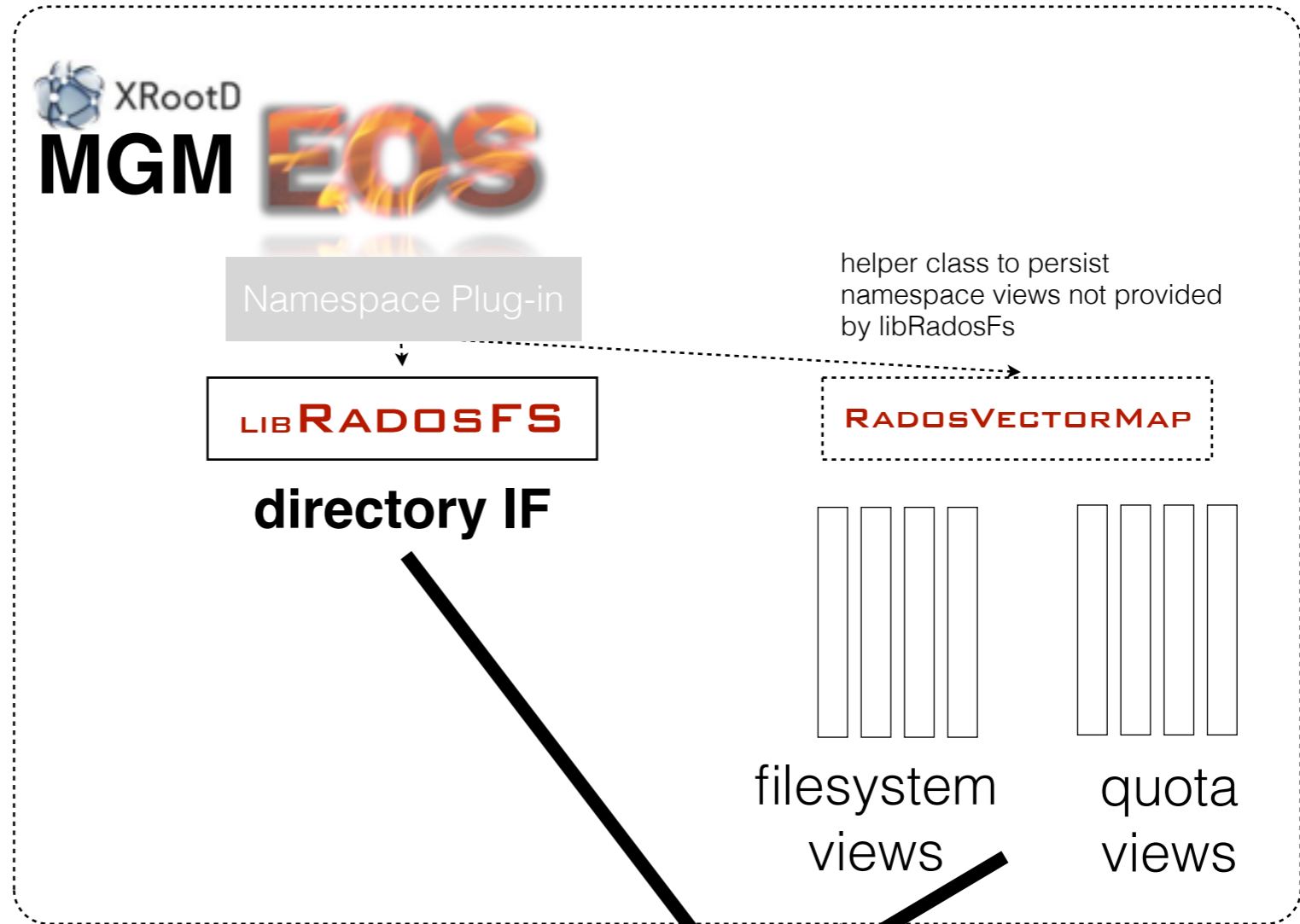


LIBRADOSFS



integration into EOS

on roadmap for 2015



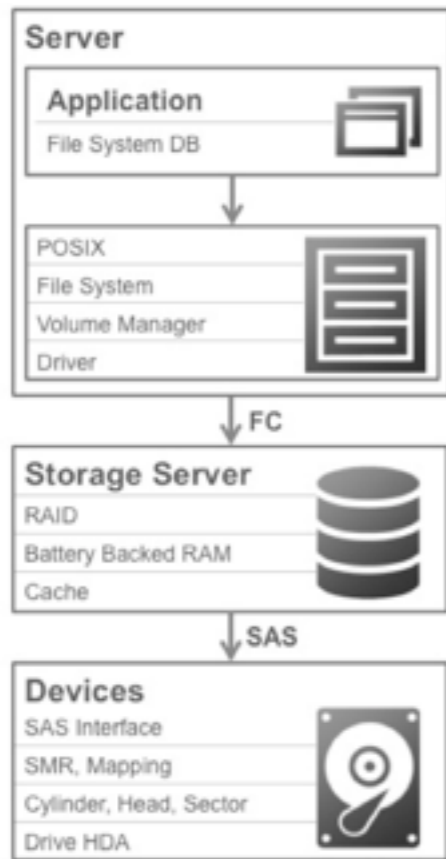
- + better scalability
- + no namespace boot time
- + many stateless MGMs
- higher latency
- slightly more complex - requires now also Ceph



SEAGATE KINETIC

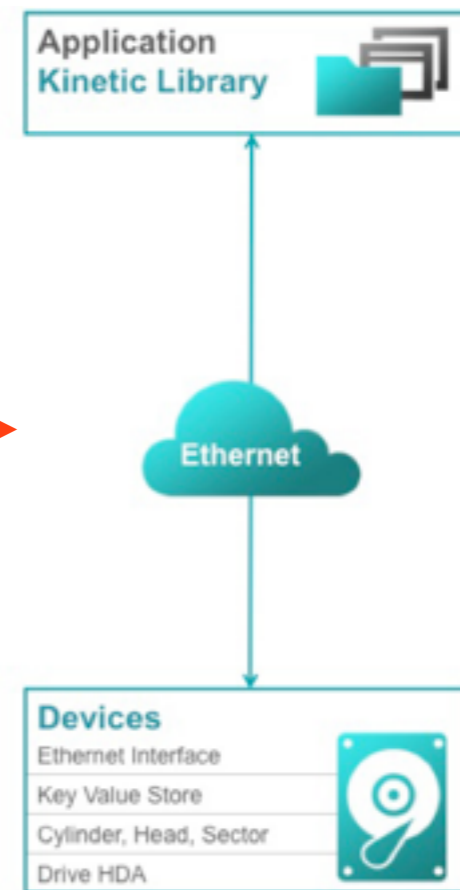
ethernet drives

conventional storage system



► POSIX

kinetic open storage platform



► Kinetic API

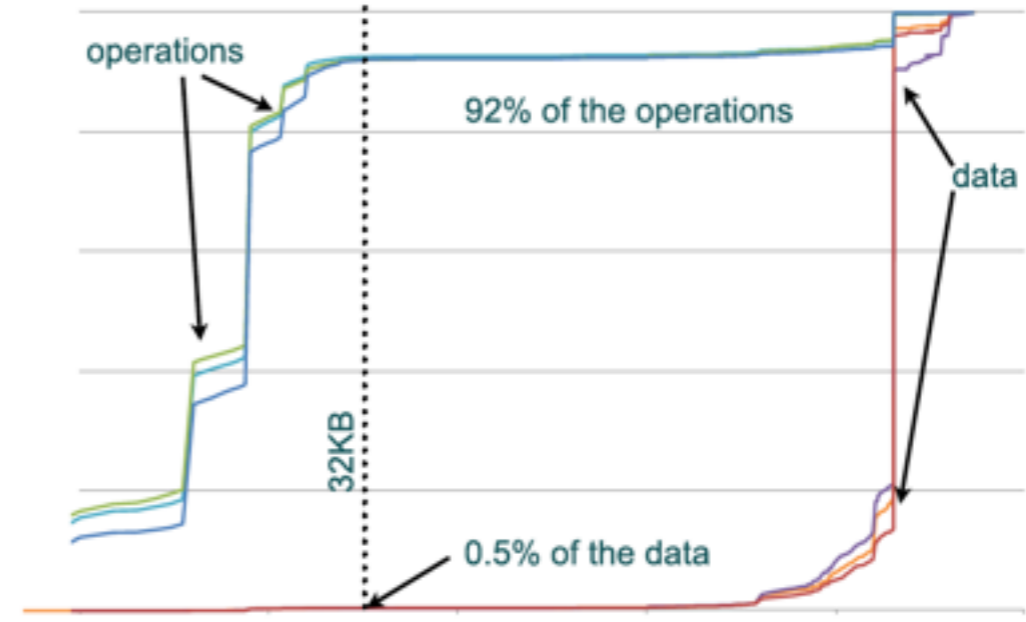
alternative approach HGST drives: provide OS on drive - no integrated remote access protocol

SEAGATE KINETIC

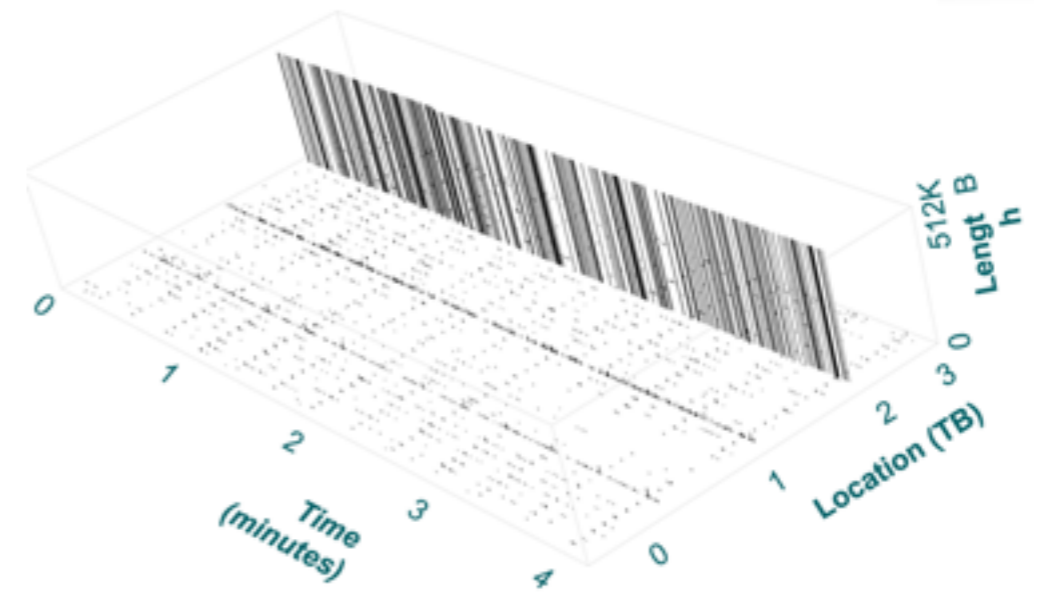
▶ why kinetic technology?
 ▶ fits technology of shingled disks
 ▶ better random write
 ▶ less meta-data overhead
 ▶ lower TCO

▶ performance expectation
 ▶ random/sequential write, sequential read:
 50 MB/s for 1M objects
 ▶ random read -15% to traditional drives
 ▶ ~ 1000 random write OOPS

▶ integrated by
 ▶ **swift**
 ▶ access via gateways
 ▶ **ceph** under development
 ▶ OSD code runs partially on disk
 difficult job with low mem/CPU



example of traditional IO inefficiency



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 cluster 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**

LIBKINETICIO

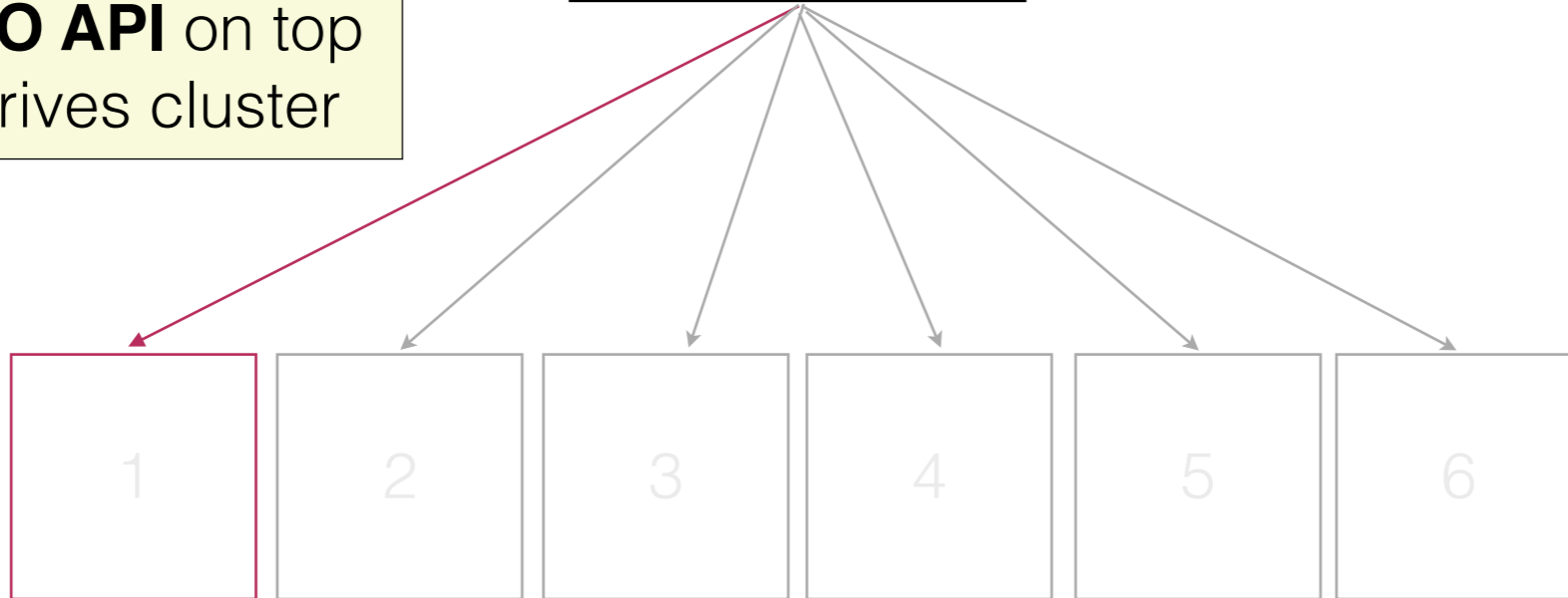
OPENLAB R&D PROJECT



libkineticicio will provide a simple **file IO API** on top of a kinetic drives cluster

LIBKINETICIO

organize disks in kinetic cubes



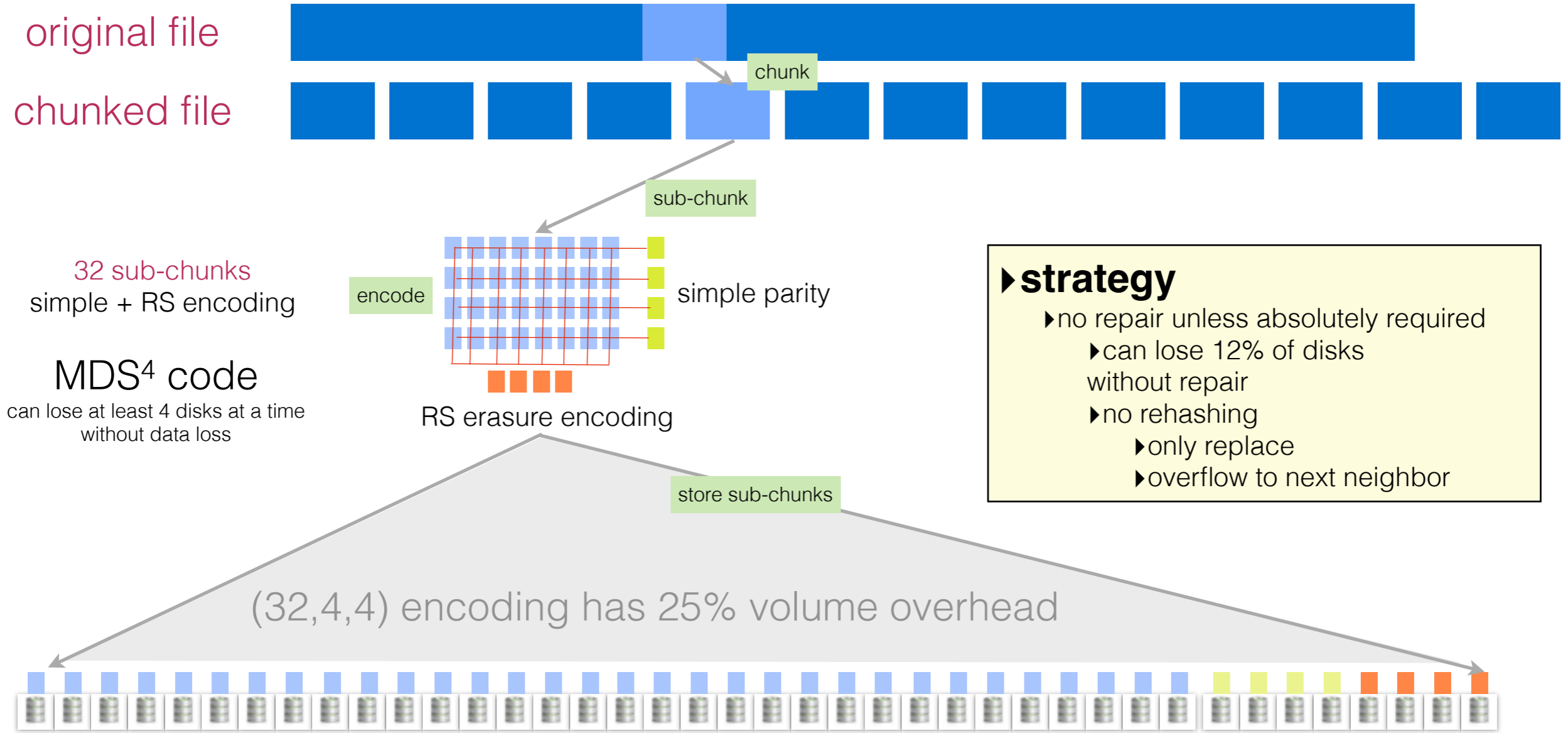
- ▶ use hash ring inside each cube for placement
 - ▶ each cube manages redundancy internally
 - ▶ file fragments are erasure encoded
- ▶ storage index for cube selection - more flexibility - better scalability



latest kinetic drive generation
252 disks - 2 PB = 15U



example of file encoding

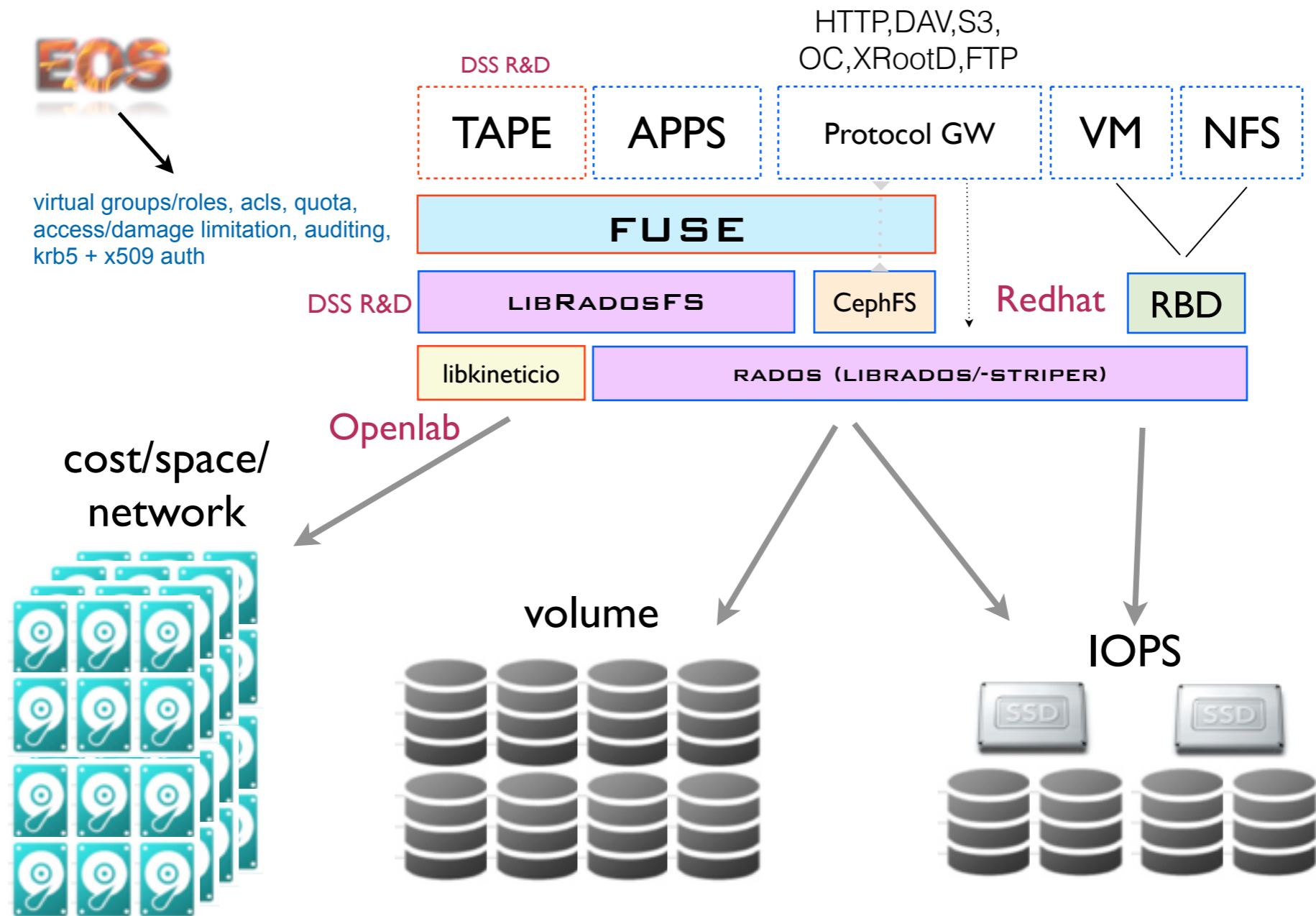


best kinetic performance for 32M chunks = 1M sub-chunks

FUTURE STORAGE VISION



MY PERSONAL VIEW & HYPOTHETICAL

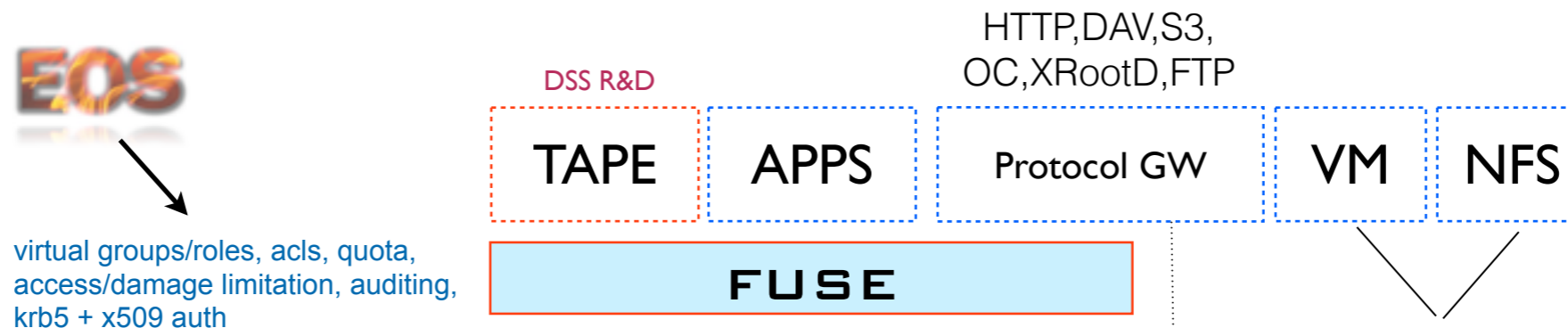


- ▶ can build whole storage infrastructure on top of kinetic & rados backends
- ▶ use generic protocol GWs on top of FUSE and rados layer
- ▶ fuse layer selects implementation between full-POSIX or POSIX-like volume storage

FUTURE STORAGE VISION



MY PERSONAL VIEW & HYPOTHETICAL



new write-back cache in FUSE

FUSE Perf. in Mb/s Linux 3.15 Miklos Szeredi			
4k	128k	test	# of memcpy's
1030	1140	native	1
80	750	fuse (direct_io,splice_read)	1
80	600	fuse (direct_io)	2
500	520	fuse (writeback_cache,splice_read)	2
320	320	fuse (writeback_cache)	4
75	470	fuse (big_writes,splice_read)	2
75	360	fuse (big_writes)	3

SUMMARY & OUTLOOK



- **object storage and ethernet drive technology** allows to build a **modular, future proof & scalable storage ecosystem**
 - ▶ customize **IOPS vs. volume vs. cost** for each use case
 - ▶ customize **access protocol & service** according to demand using widely used open source building block
- CERN storage R&D aims to contribute **generic & reusable libraries** to efficiently use the ceph and kinetic storage environment
 - **LIBRADOSFS** to build scale-out meta-data services
 - **LIBKINETICIO** for low-cost reliable file storage
- CERN DSS already made **significant upstream contributions** to the ceph project
 - **libradosstriper** - large object striping library
 - intel **ISA-L** erasure coding plug-in library
- R&D aims to contribute **non-HEP specific storage building blocks** keeping flexibility for required CERN customization

Thank you!



Questions or Comments?

SOFTWARE

R&D REPOSITORY (UNSUPPORTED)



disclaimer: these are R&D repositories - the software comes with ABSOLUTELY NO WARRANTY and we currently don't guarantee functionality and long-term support

libradosfs GIT repository

<https://github.com/cern-eos/radosfs>

XRootD integration of libradosfs

Diamond bundle

<https://github.com/cern-eos/eos-diamond>

Documentation

<https://github.com/cern-eos/eos-diamond/wiki>

Repositories

<https://eos.cern.ch/rpms/eos-diamond>

<https://eos.cern.ch/rpms/eos-diamond-testing>

file-only use cases: XRootD 4.2 CEPH file plug-in preview (not yet released):

<https://github.com/xrootd/xrootd/tree/master/src/XrdCeph>