A Spotlight on **IO Server** solutions for LHC physics

Dr. **Andreas-Joachim Peters**
CERN IT-ST

IO overview

- **Introduction**
  - seven aspects of IO for high-throughput applications

- **Server technology perspective at CERN**
  - near term outlook for HEP at CERN
  - Server R&D at CERN
  - XRootD

- **Server evolution and trends**
  - Technologies for NVMe and hyper converged storage
    - NVMesh
    - DAOS
    - SkyhookDM

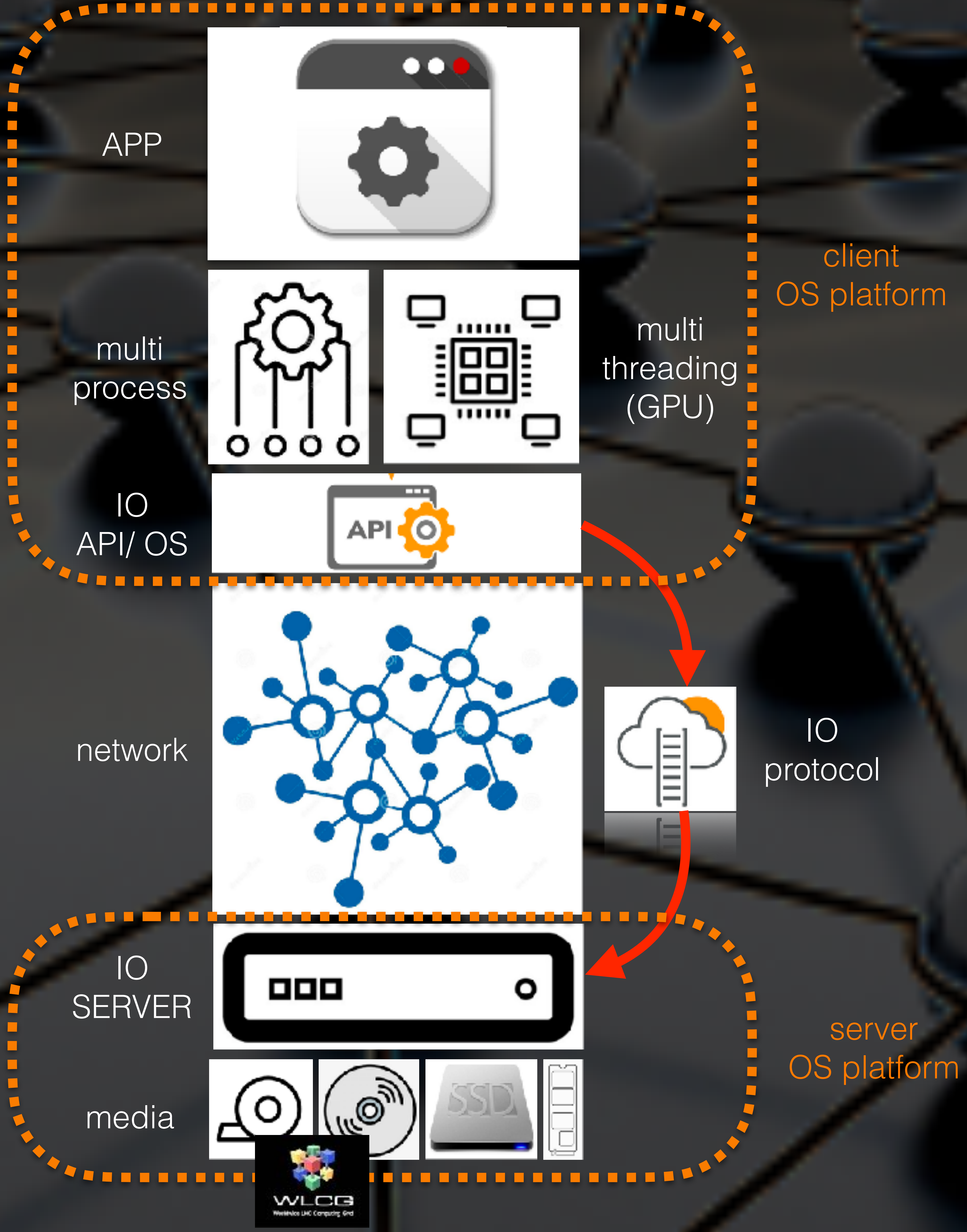
Introduction

Preface:

IO use case of HEP is still simple.

Aspect 1
removing bottlenecks

IO stack



There are always several ways to solve a given problem !

TTree -> RNtuple
POSIX -> KV
Random -> Seq IO
10 GE -> 100 GE
NFS -> DAOS
RADOS-> NVMesh
provide more HW …

Question

Where do we solve IO problems?
Where are IO systems evolving?


Aspect 2
platform
IO platform
client server
client
platform
server
platform
Realm of Disk & Tape
exploit volume scaling
WLCG
Worldwide LHC Computing Grid
hyper converged
client=server
Realm of Flash
exploit IO locality

Aspect 3
tiering

IO in tiered storage

transparent cache

PERF tier

access driven migration

VOLUME tier

managed cache

PERF tier

explicit migration

VOLUME tier

WLCG
Worldwide LHC Computing Grid

Question
When does Storage Tiering actually make sense?

Aspect 4
stream types

IO Server for Data Transfer | Access

we require two types of IO server

data transfer

Storage Site 1

WAN

high bandwidth
RW Streaming

Storage Site 2

data access

Application

LAN [WAN]

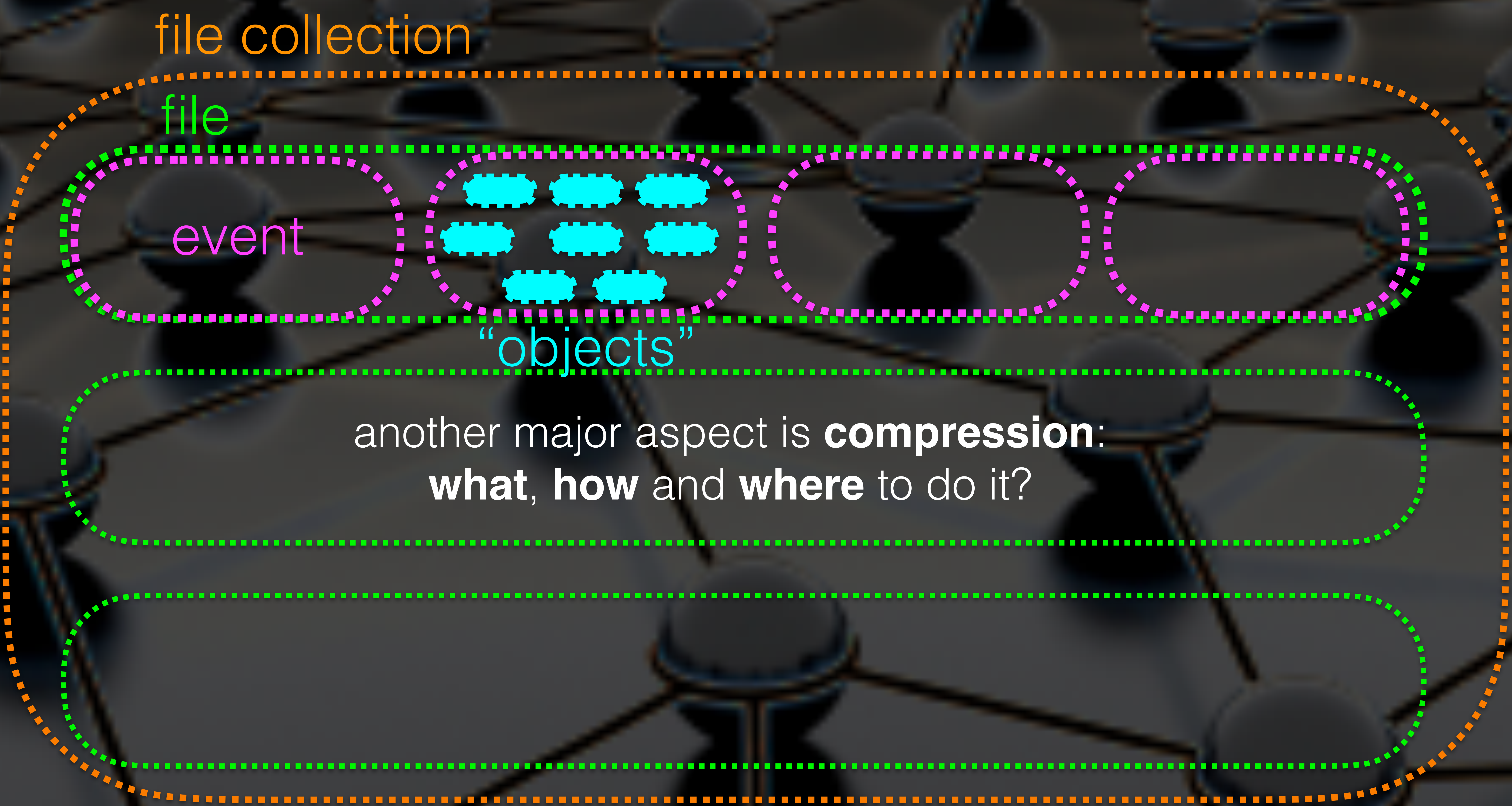
RO Streaming + Random
WO Streaming

Storage

Diversity or Zoo?
How many server/protocol solutions does one really need?

Aspect 5
data format

IO data abstraction in LHC physics

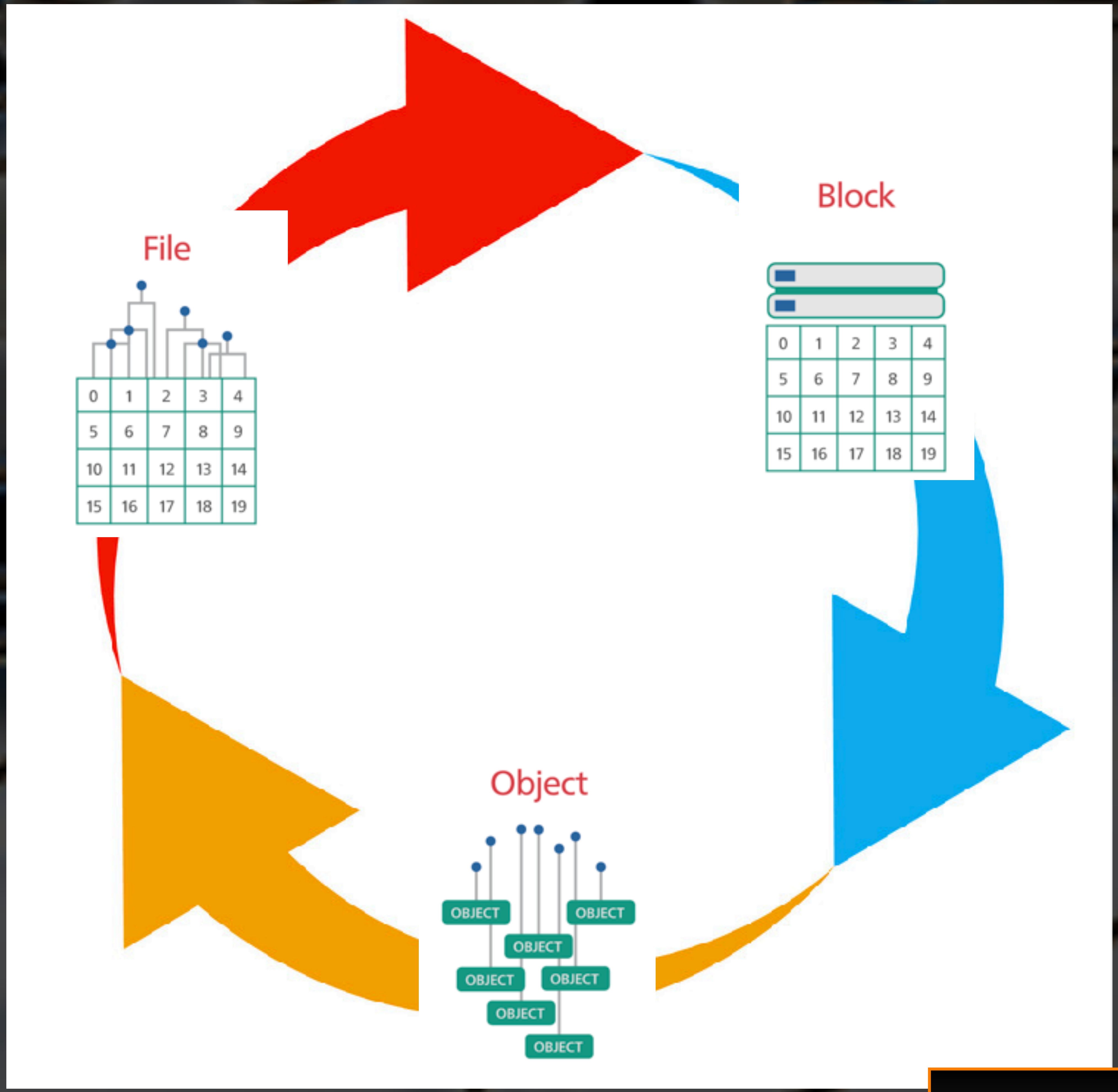


Given Facts / Questions

1 The **storage system** is in general **not aware of** the the **data structure** (?)
2 We always decompress on application side (?)

Aspect 6
API & storage

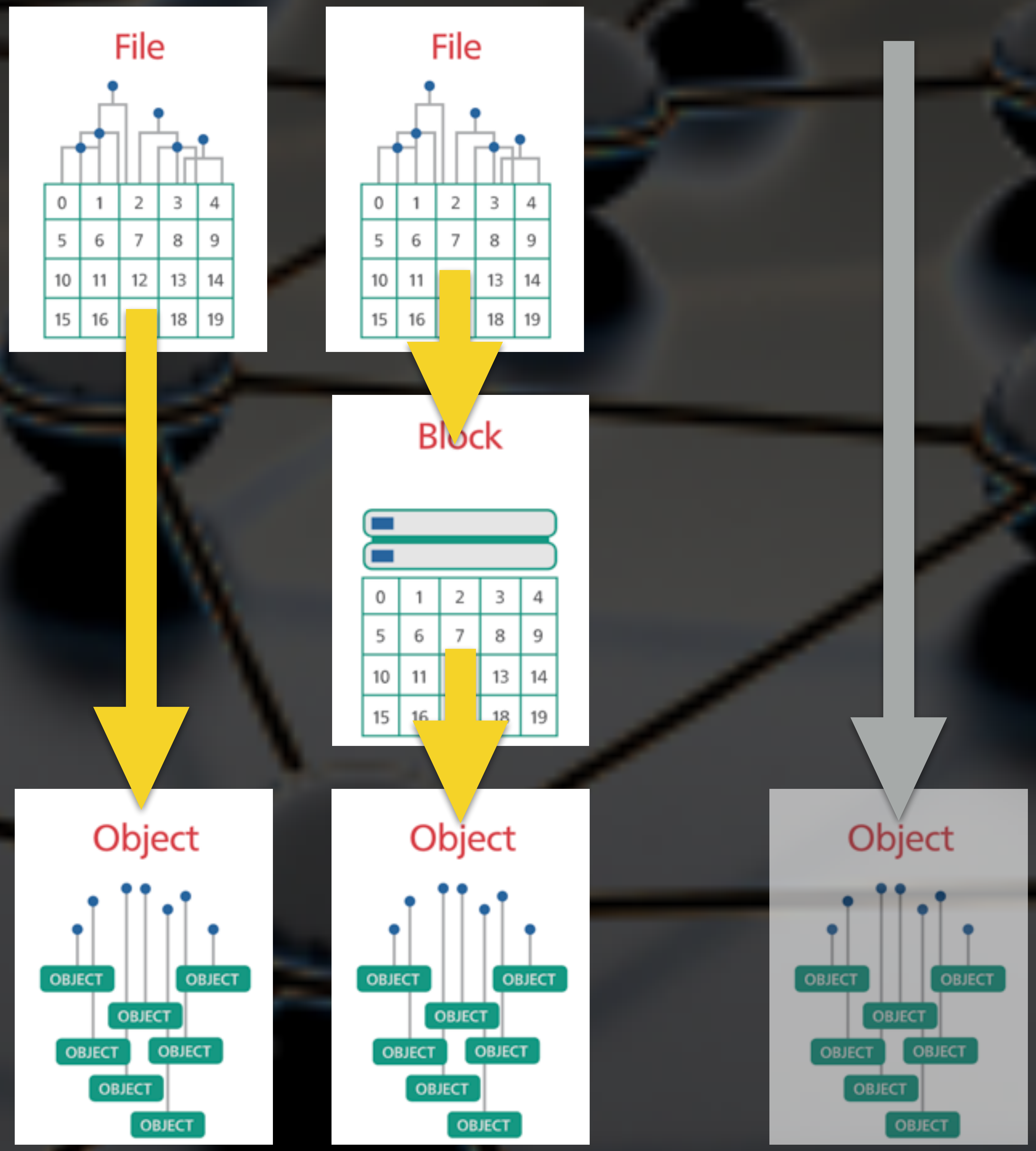
IO Object Storage



FILE api common dominator

FILE api must not imply to follow POSIX !!!

file = collection of objects



Simplify your life

We can stick to FILE APIs even when we use object storage directly

Aspect 7
storage media
IO media
a certain vendor's view . . .
media
price examples
250x
±4000 Euro/TB
500x
±8000 Euro/TB
75x
±1200 Euro/TB
4-6x
±100 Euro/TB
0.5-1x
8-16 Euro/TTB
DRAM
HOT TIER
intel OPTANE DC
PERSISTENT MEMORY
intel OPTANE DC
SOLID STATE DRIVE
Intel® QLC 3D Nand SSD
HDD / TAPE
COLD TIER
WLCG
Worldwide LHC Computing Grid
intel INTEL OPTANE SSD DC P4800X SERIES
intel
meta
data
data

Aspect 7
storage media

IO media

capacity comparison for a given budget with no redundancy

media
price examples

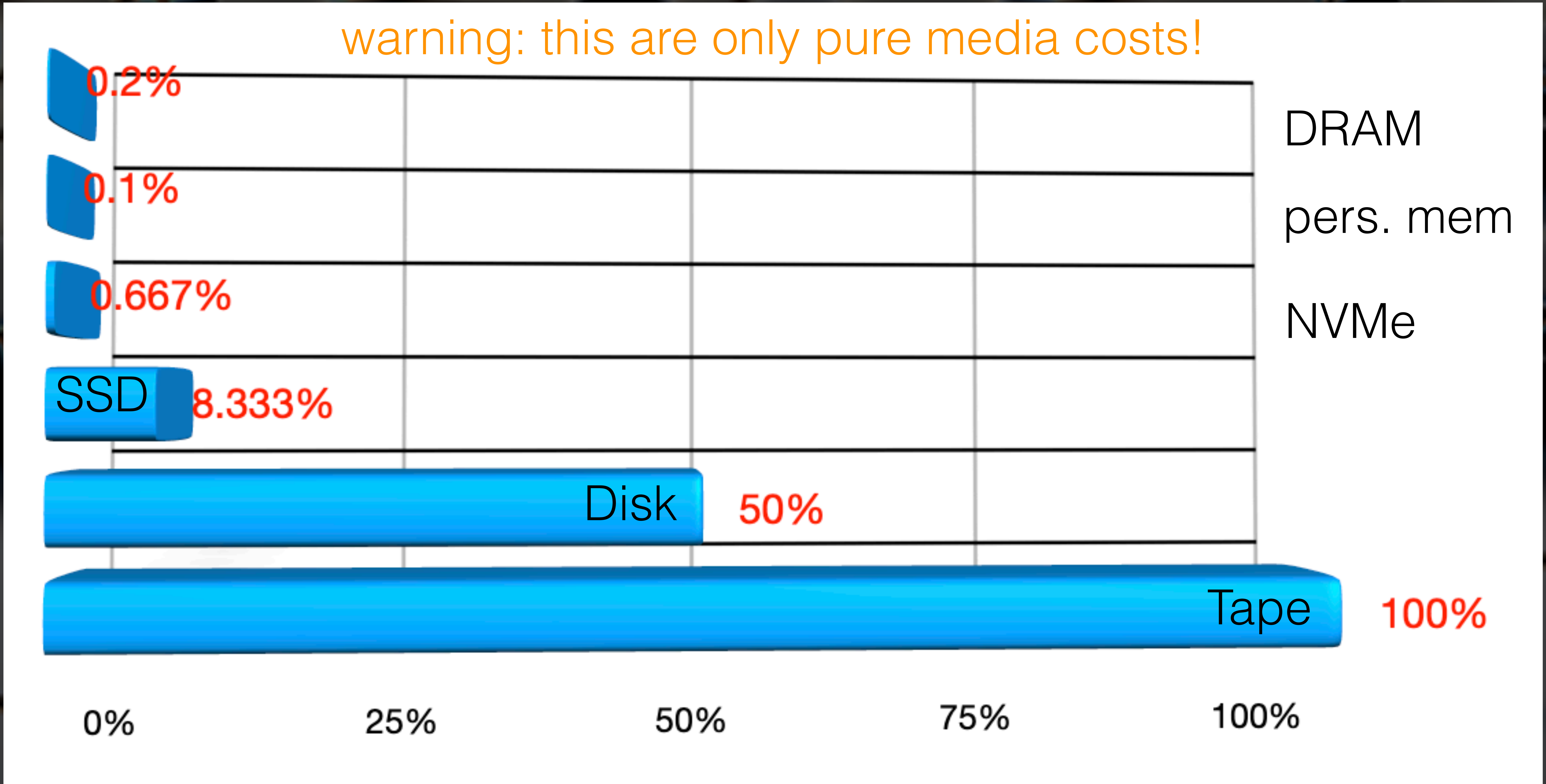
250x
±4000 Euro/TB

500x
±8000 Euro/TB

75x
±1200 Euro/TB

6x
±100++ Euro/TB

0.5-1x
8-16 Euro/TB



IO Server Perspective at CERN

IO perspective at CERN for Run 3

HDD **Baseline**

assume a **disk based** storage platform as baseline service when developing data formats and application IO

- **disks** (+ tapes behind … see CERN Tape Archive project CTA)

SSD **NVMe**

are great ***local*** or ***remote-local*** low-latency high-bandwidth temporary, caching or burst-buffer storage

- **batch job stage-in, stage-out etc.**
- great for end-user analysis

Improvements by Hardware & **Software**

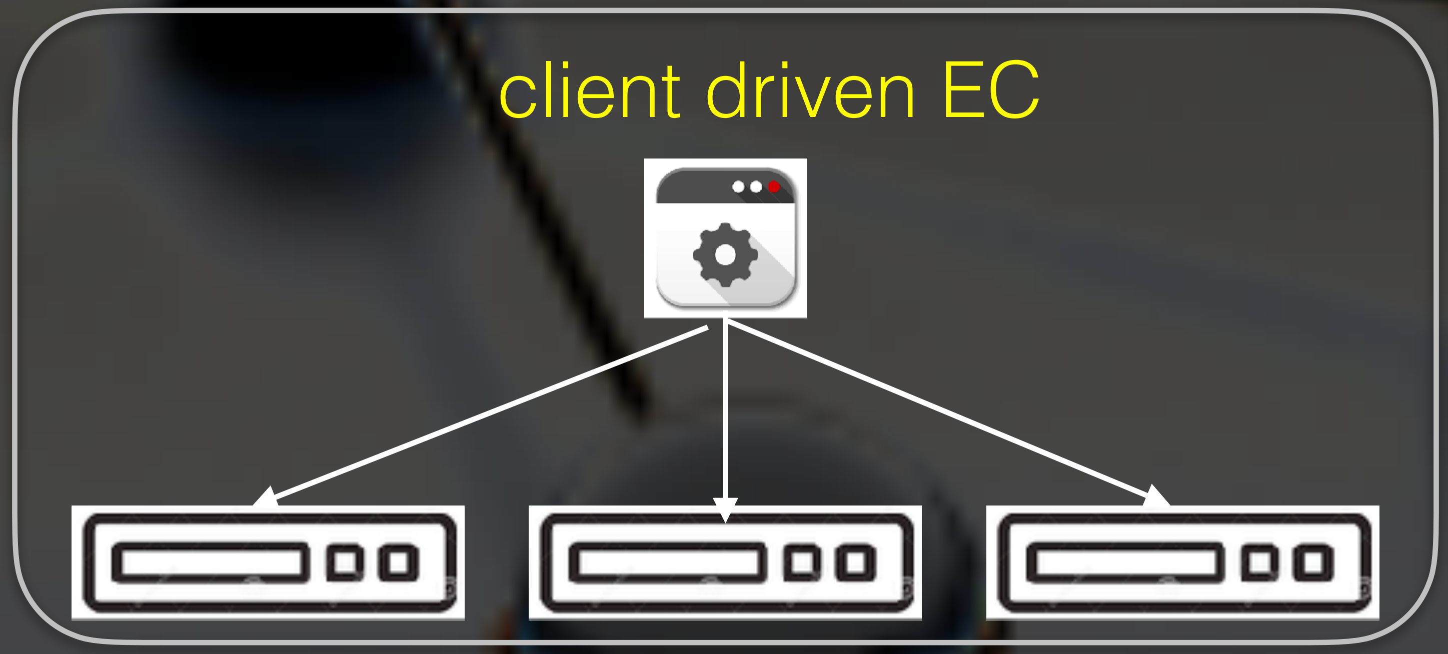
100GE + **PCIe** boost sequential streaming performance using HDDs using **parallel IO** (Object Storage, RAIN, Erasure Coding …)

- high-bandwidth HDD streaming allows easy inclusion of SSDs/NVMe into workflows to improve performance [if needed]
  - not all LHC use-cases need NVMe (high IOPS) to be efficient
  - straightforward to get 1-2 GB/s streams per file - are our applications requiring more?

CERN key technologies for Run 3

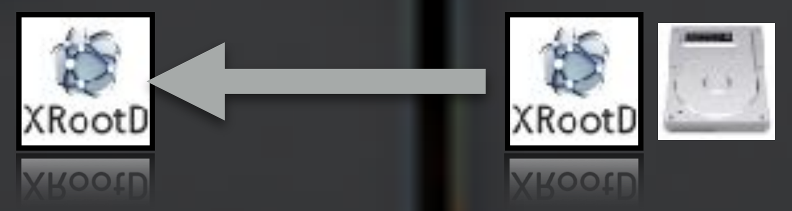
100GE/PCIe
allows factor 4x-10x improvements for streaming IO out of one disk server

(client-driven) **erasure encoding**
high-bandwidth streaming for storage tiering at low cost with HDDs



XRootD Server
modular storage for WAN + LAN Transfers/Access + possibly Caching

all physics IO:
XRootD + JBOD server



IO Server - EOS@CERN today

physics data

Number ...	Number ...	Total Sp...	Used Sp...
7.012 Bil	384 Mil	344.05 PB	255.13 PB

Current ...	Current ...	IOPS	Write Throughput	Read Throughput
8 K	137 K	436 K	22.2 GB/s	112 GB/s

IO Server - CERN Run-3

physics data

General **Direction**

cost driven approach “provide IO bandwidth + volume at low cost - (less) IOPS - use big files”

Storage **Capacity**

will exceed **500 PB** raw disk space this year

- +**134 PB** additional capacity for physics (9600 HDDs)
  - **14 TB** disks / **96** per server
  - **100** GE per server - no blocking factor

low-cost high performance **Storage**

100 PB pool for ALICE experiment: **O2**

- 7000 HDDs, connectivity 75 x 100GE, EC RS(12,10)
- nominal 2 TBit streaming IO with client driven erasure encoding

IO Server - CERN Run-3

physics data

- **NVMe** or **SSDs** are not used for Big Data Storage
- dedicated analysis facility with local **NVMe's**+hyper converged CephFS (HPC)

- **HDD** price still factor 5 better, power consumption HDD/SSD not significant extra cost at CERN

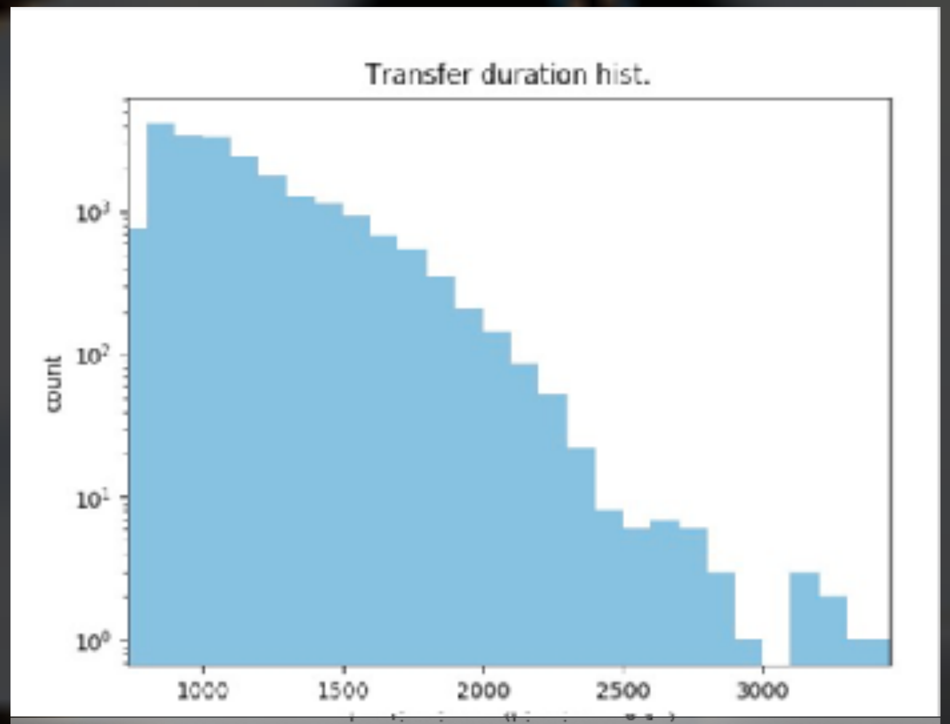
- **TAPE** still cheaper than disk in large installations
  alternative approach : EC disks as tape replacement e.g. KISTI project

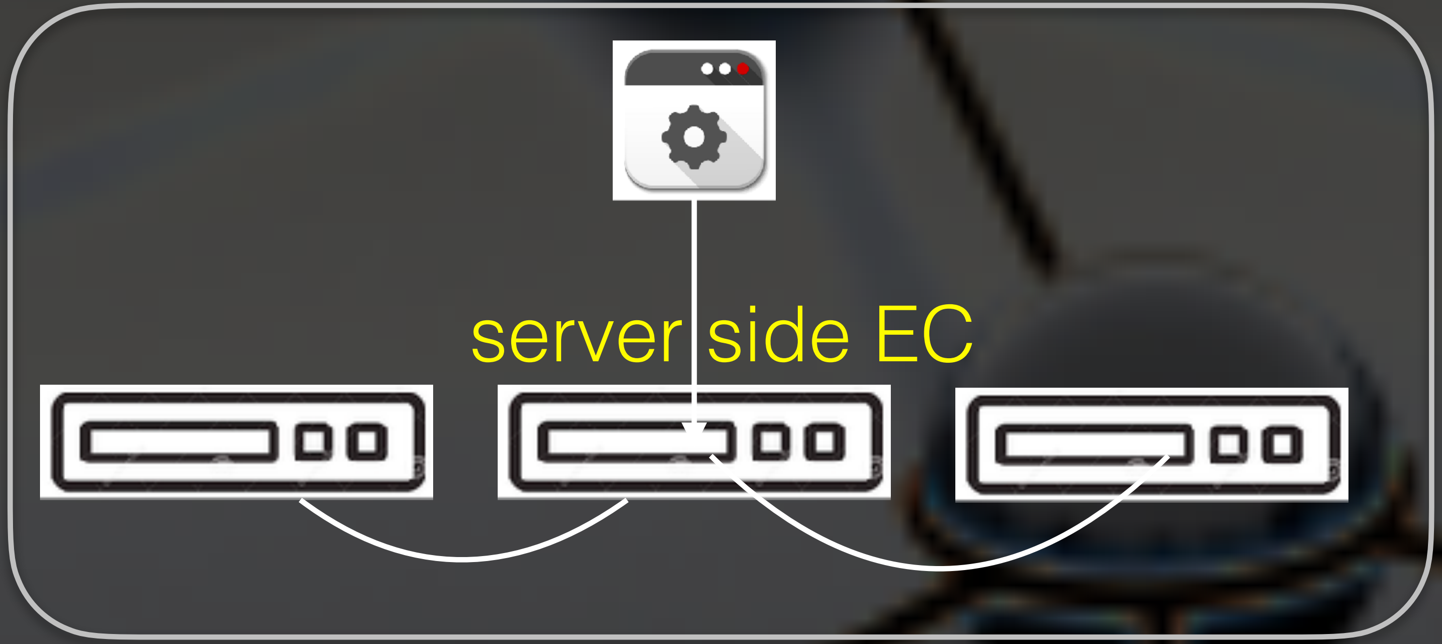
- **major part** of IO volume we serve is essentially **sequential** (upload/download or few initial seeks + forward reading/seeking) access

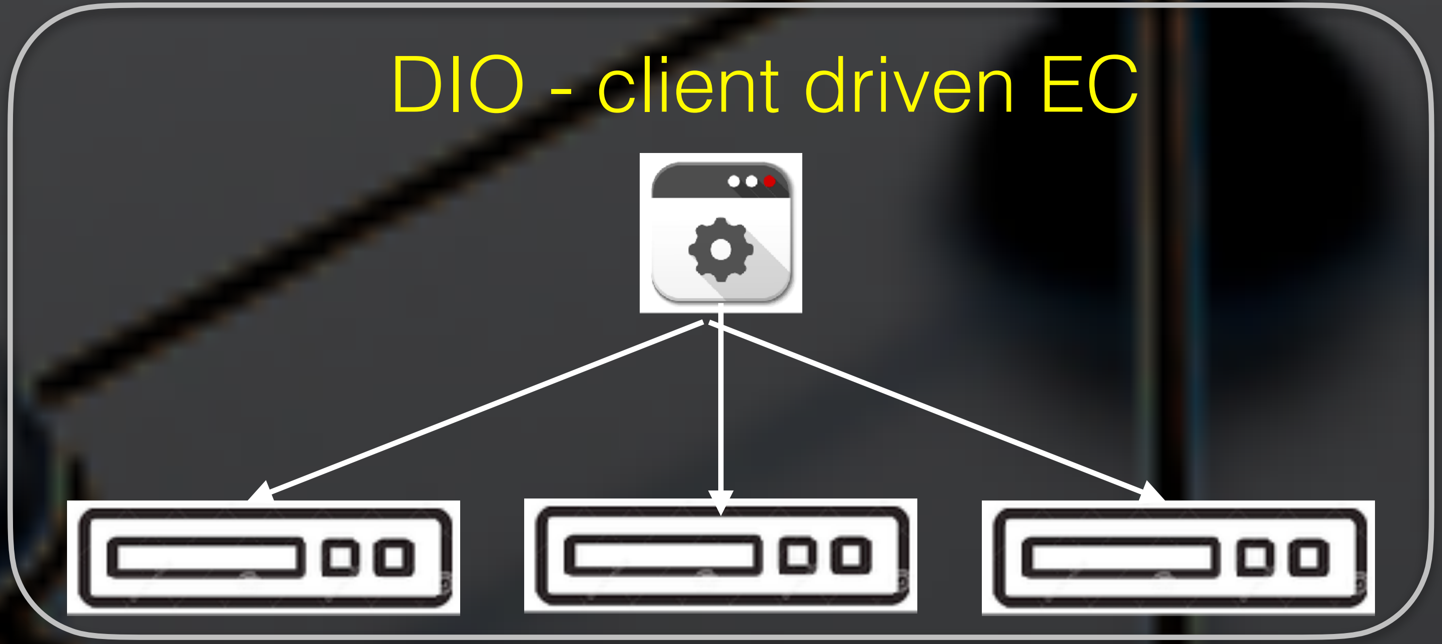
EOS **DIO**
Client side EC

IO Server - **CERN** R&D

- **O2** use case requires write performance guarantees - fixed time window
  - until today EOS supports only client side EC (parallel IO) for reading
- evaluating XrdEc plug-in for client side EC (parallel IO) for reading and writing optimised for streaming (supports random reads) - server-side EC (see RADOS) yields traffic amplification ~2x
  - less tails in transfer durations - achieved stream performance >1.8 GB/s [RS(12,10)]







5. 3 data servers, 300 streams (15GB/s of aggregate throughput), 1 hour run

Avg transfer duration = 1151.5723889783164 msec
Avg. transfer rate = 1.8447392182702522 GB/s
Median data transfer rate: 1.8726591760299625 GB/s
Median transfer duration: 1068.0 msec
Data transfer rate standard deviation: 0.41908173851372O5
Transfer duration standard deviation: 309.24501469327305

IO HEP exotic server/protocol solution

- **XRootD** has become a widely adopted framework for IO in HEP over the years
  - multithreaded C++ server using TCP/IP
    - **federation/clustering** capabilities
    - storage back-end **plugins** (POSIX, hdfs, ceph)
    - **proxy cache** (block oriented) *xCache*
    - **third party transfers** (core functionality for a tiered storage architecture
    - protocol plug-ins **root & http(s)** protocol
      - building block to integrate many storage systems into the global infrastructure

XRootD

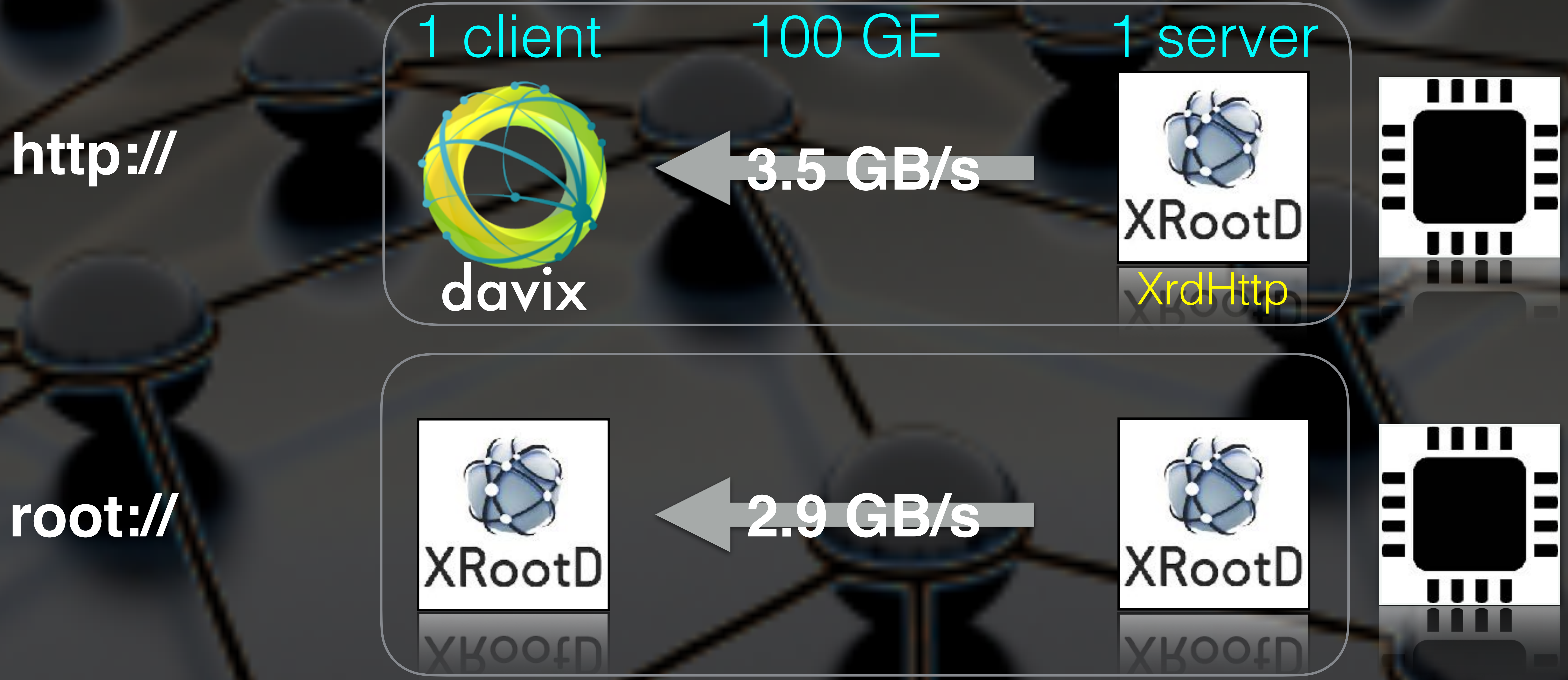
xrootd.org

Questions

- is it good enough for **100GE** networking?
- is it good enough for **NVMe** based storage?

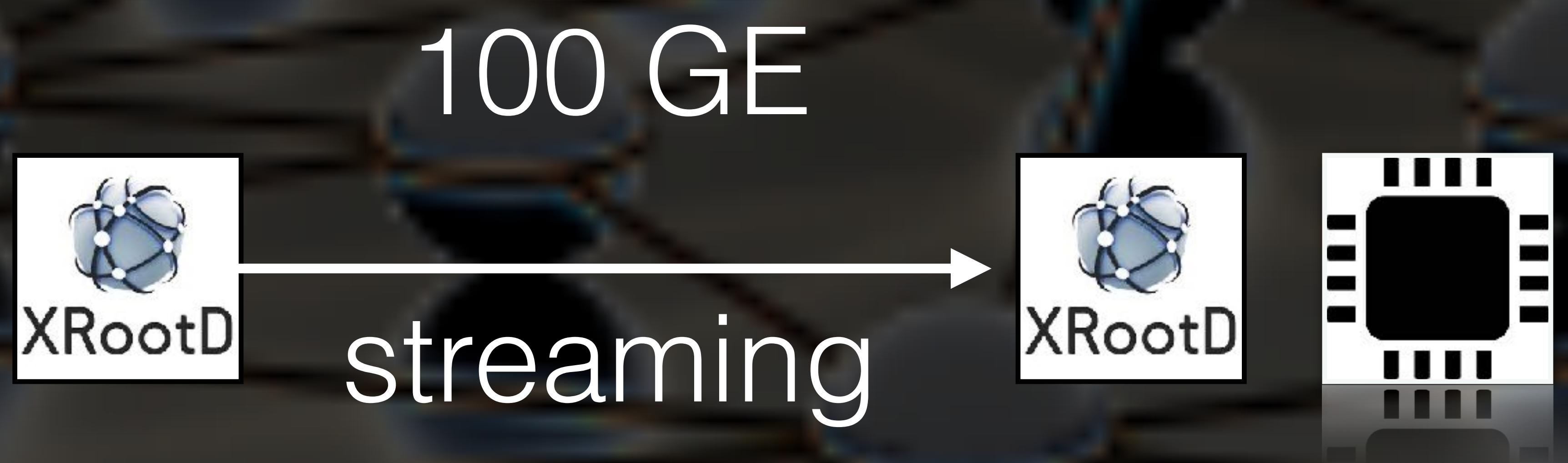
IO 100GE XRootD

single TCP/IP stream performance

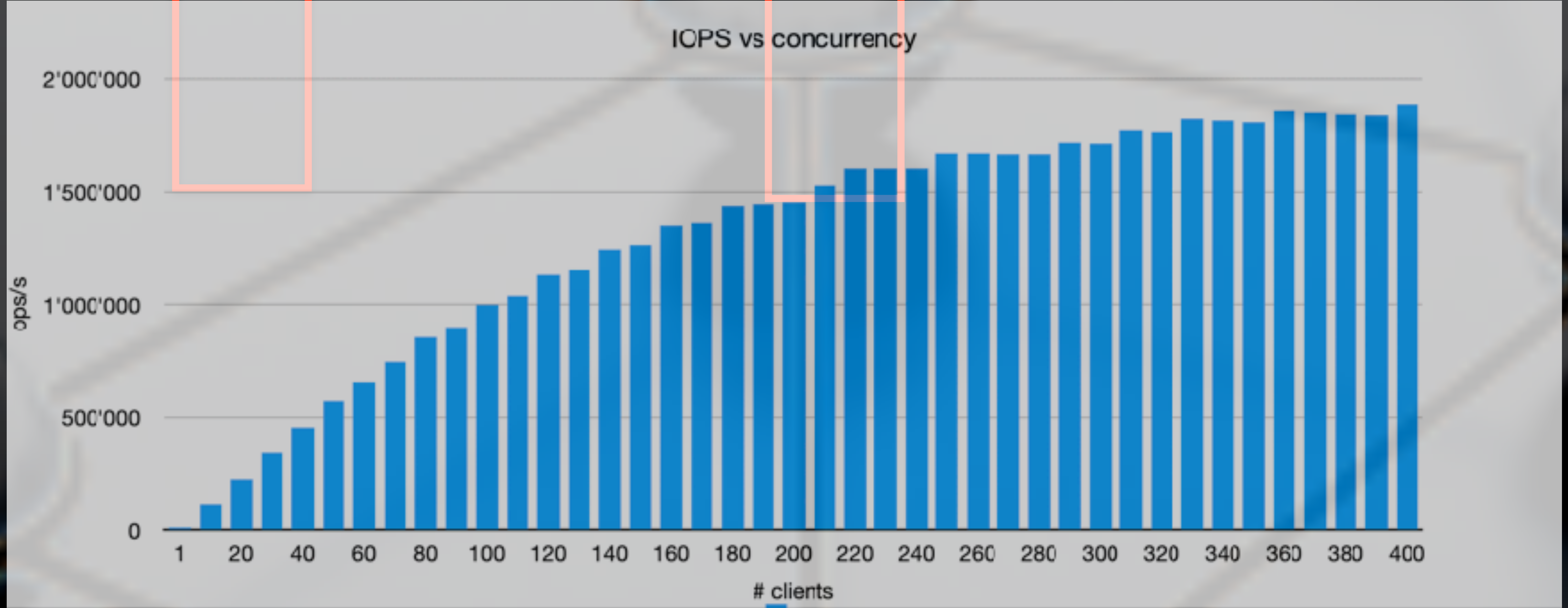


no data encryption

100 GE Technology IO Server - CERN now



usr	sys	idl	wai	stl	read	writ	recv	send	in	out	int	csw
4	7	89	0	0	0	64k	8674M	9000k	0	0	208k	127k
5	11	83	0	0	0	40k	11G	12M	0	0	294k	196k
6	11	83	0	0	0	56k	12G	13M	0	0	277k	224k
6	10	83	0	0	0	48k	11G	13M	0	0	266k	211k
6	11	83	0	0	0	32k	11G	13M	0	0	264k	206k
6	10	84	0	0	0	8188B	11G	13M	0	0	269k	219k
6	10	84	0	0	0	48k	11G	12M	0	0	257k	205k
6	10	84	0	0	0	16k	12G	14M	0	0	286k	236k
6	10	84	0	0	0	104k	12G	13M	0	0	284k	233k
5	10	84	0	0	0	100k	11G	13M	0	0	272k	224k
6	10	84	0	0	0	24k	12G	13M	0	0	270k	221k
6	10	84	0	0	0	44k	11G	13M	0	0	260k	219k
5	10	85	0	0	0	132k	11G	13M	0	0	293k	232k
6	10	83	0	0	0	24k	12G	14M	0	0	277k	226k
6	10	84	0	0	0	80k	12G	14M	0	0	278k	237k



client

10 streams 12 GB/s

16% CPU

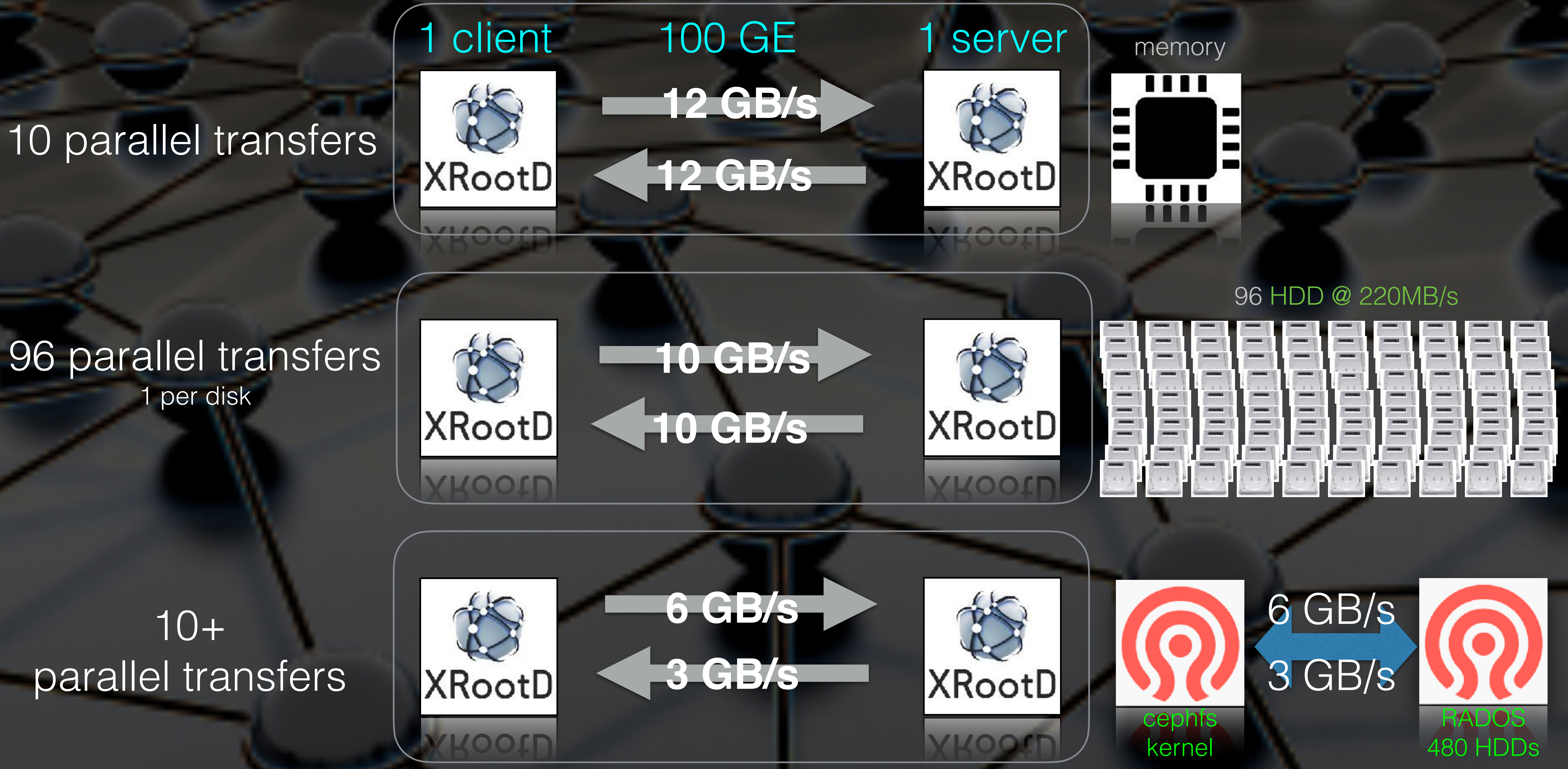
IOPS limit ~ 2M

read from BC single file

but only 40k from

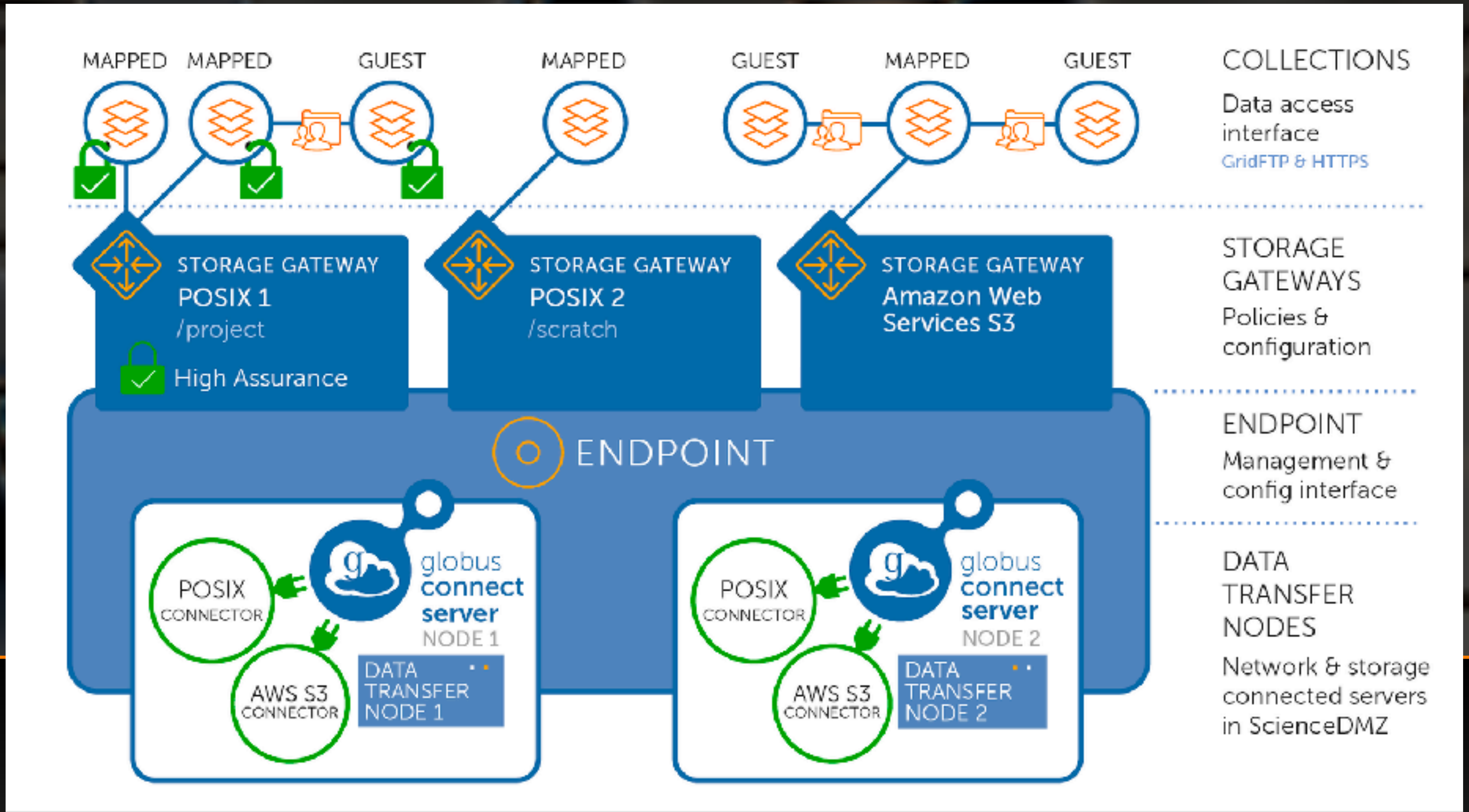
NVMe with 550k IOPS

IO 100GE XRootD



Globus Connect **V5 Server**

Commercial
Data Transfer for HPC



- supports **gridFTP** & **HTTPS**
- backends**: Google Drive, Google Cloud Storage, POSIX, POSIX with file staging, Box, Ceph, S3, SpectraLogic BlackPearl, and iRODS**
- **HPC** sites deploy Globus Connect Server (4,5)
  - but **no support in WLCG file transfer service** (yet?)
  - **not** really **aligning with WLCG strategy** to drop GLOBUS

IO Server Evolution & Trends

IO systems

in the outside world…

- local filesystems
- local KV stores/DBs
- (adhoc)filesystems on remote devices
- parallel/distributed filesystems
- distributed KV(object) stores/DBs
- high-level object storage

RBD

EBS

DFUSE

CephFS

RADOS

S3

IO Trends

- **get rid** of POSIX for HPC (sometimes …)
- **more** Open Source Software (rados, daos, scylladb)
- **object storage** model for parallel IO
- **EC** (client-side) for economy and availability
- **FUSE** and ad-hoc filesystems on top of Object Storage (juicefs, dfuse …)
- **NVMe** + **persistent memory** as high IOPS tier
- **tiering** to moderate the price problem of NVMe

IO Server for NVMe

What`s your PROBLEM

- With the introduction of **NVMe** and **Optane Memory** it has quickly become evident, that the **asynchronous interfaces** of LINUX are **not efficient** to deal with extreme low latency (micro seconds) devices

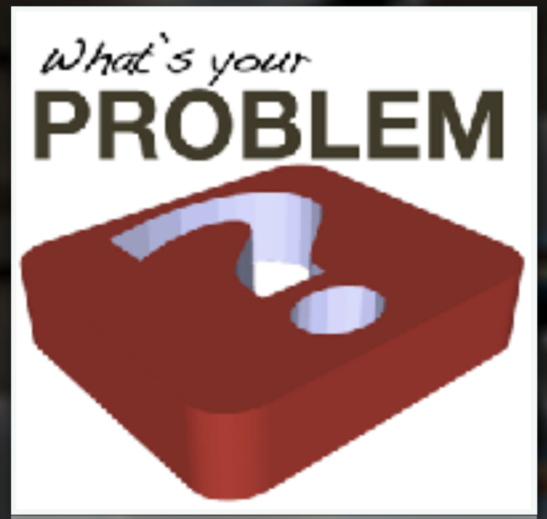
- an example:

  CEPH OSDs cannot deliver IOPS of NVMe devices over a network to remote clients with low latency
  - 80k IOPS at 6ms latency per server

    => rethink (rewrite) your IO server

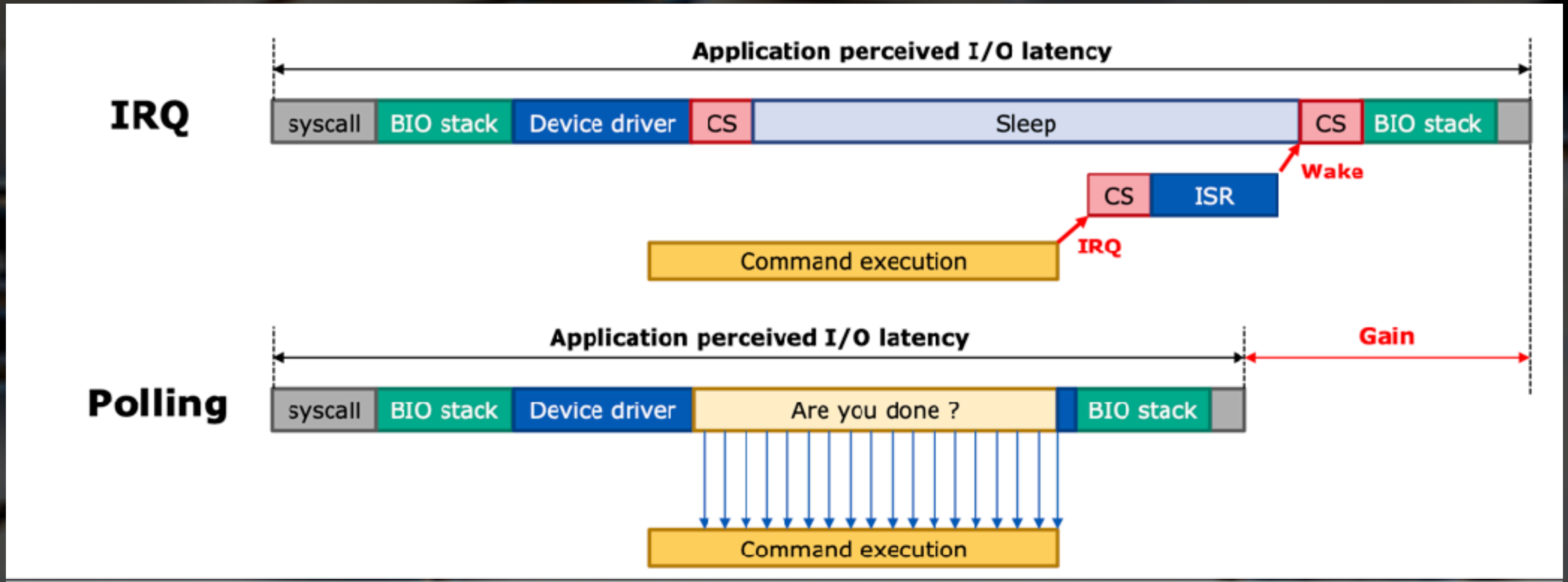
IO Server Polling

How do you get the IOPS of NVMe or Optane Memory out?

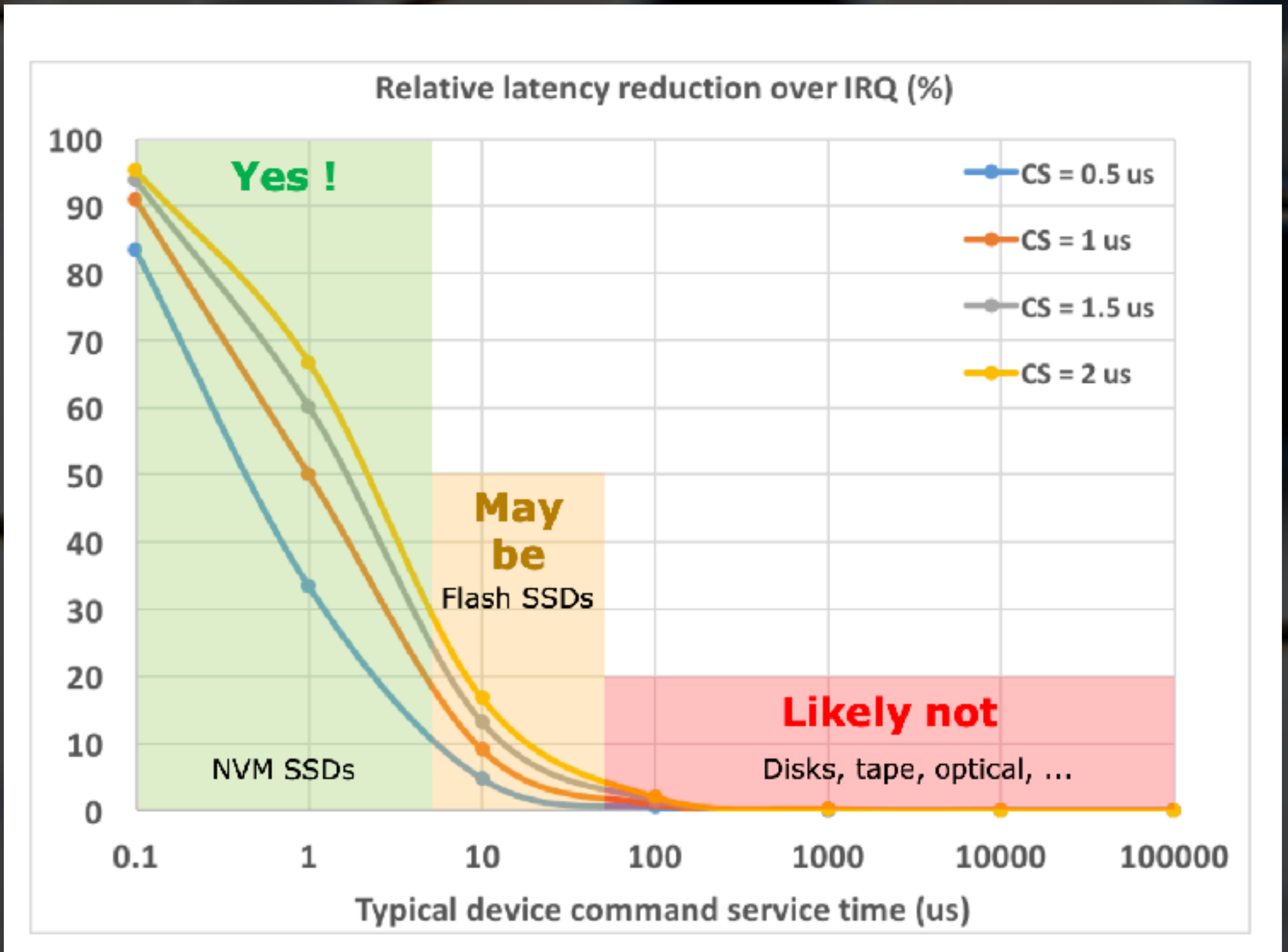


Trade-off CPU load for lower I/O latency

Polling vs. Interrupt



Where to use polling …



IO Server io_uring

How do you get the IOPS of NVMe or Optane Memory out?

https://thenewstack.io/how-io_uring-and-ebpf-will-revolutionize-programming-in-linux/

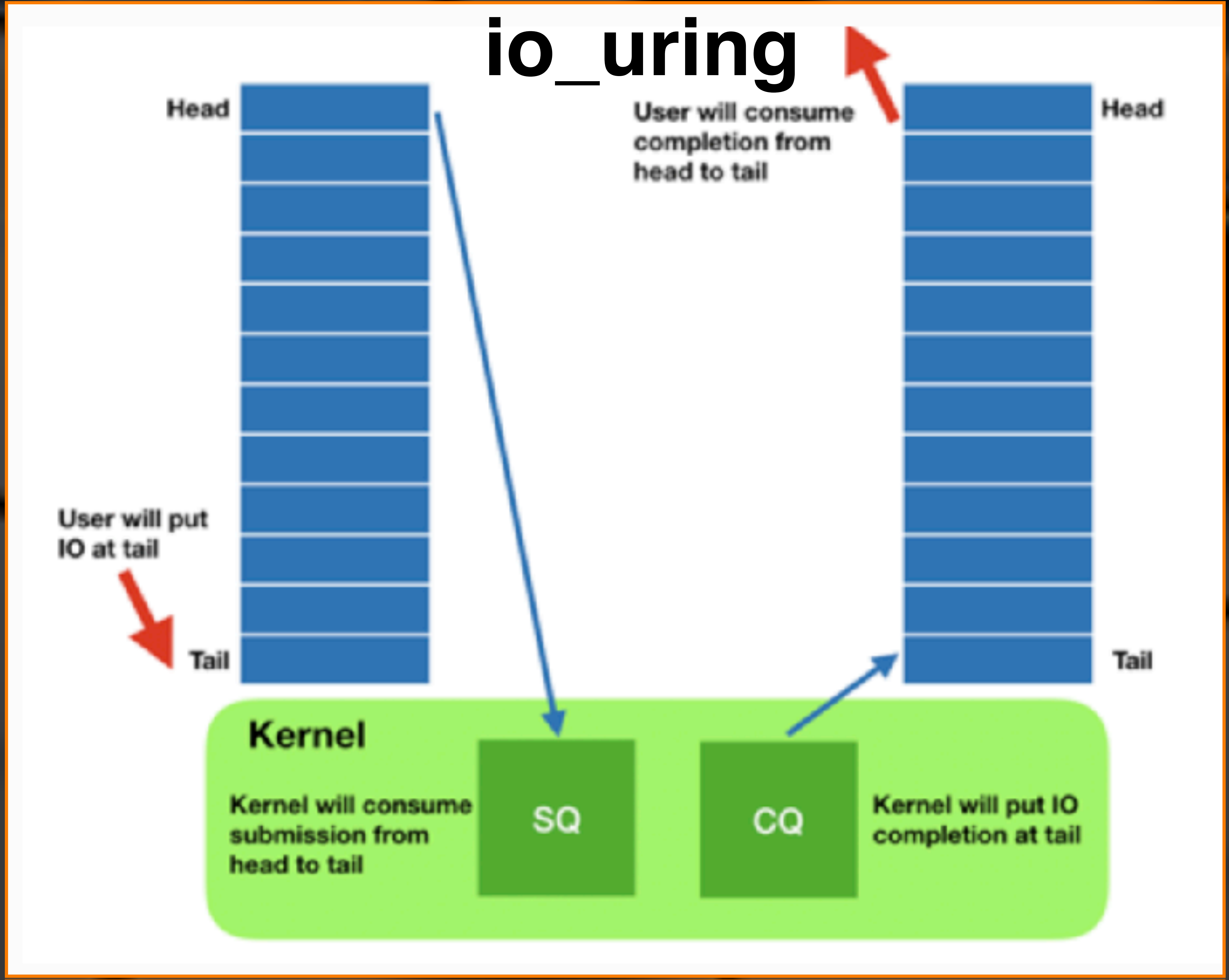
What's your PROBLEM

```
ssize_t read(int fd, void *buf, size_t count);

ssize_t write(int fd, const void *buf, size_t count);
```

Evalution of IO interface in LINUX

- sync IO
- posix AIO : thread pool running sync IO
- linux AIO : only direct IO
- io_uring : real asynchronous non blocking IO in newest kernel

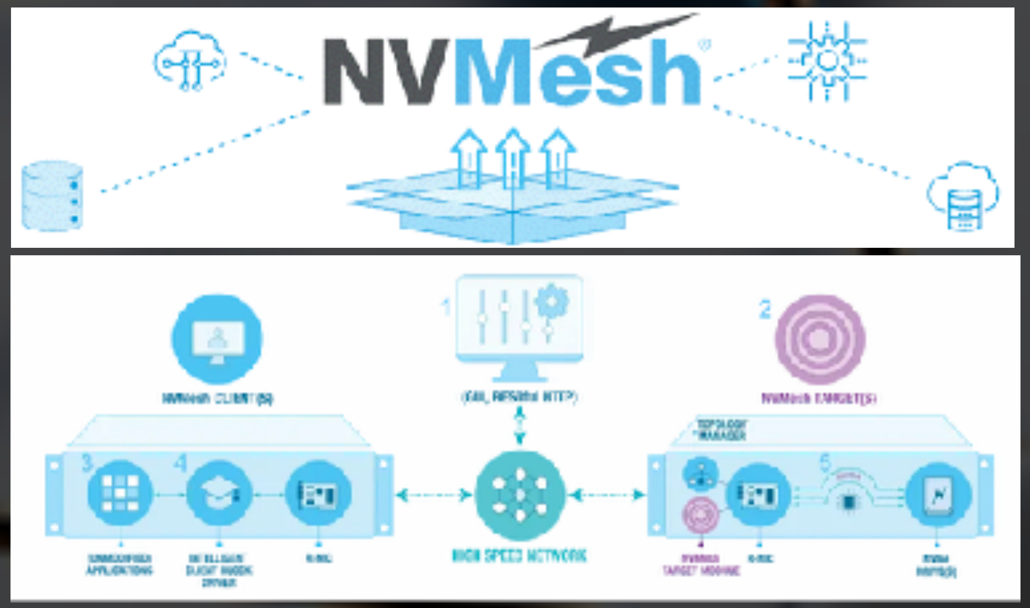


IO Server - Alternative Ways

remote direct drive access

RDDA

bypass CPU

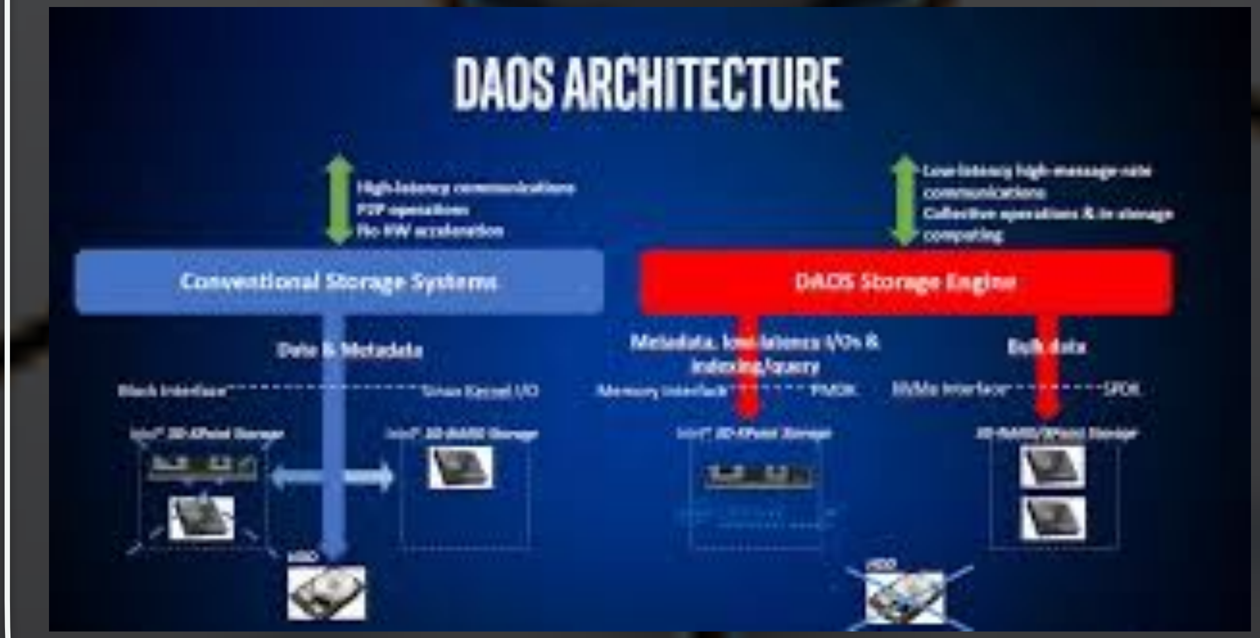


storage performance development kit

SPDK

polling instead of interrupts

daos

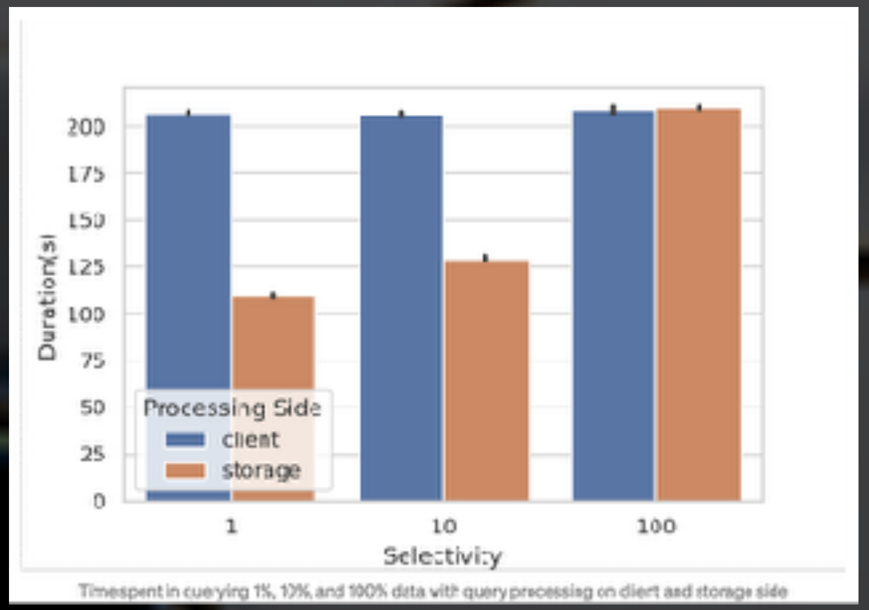


data aware storage with server processing

∅NETWORK

SkyhookDM

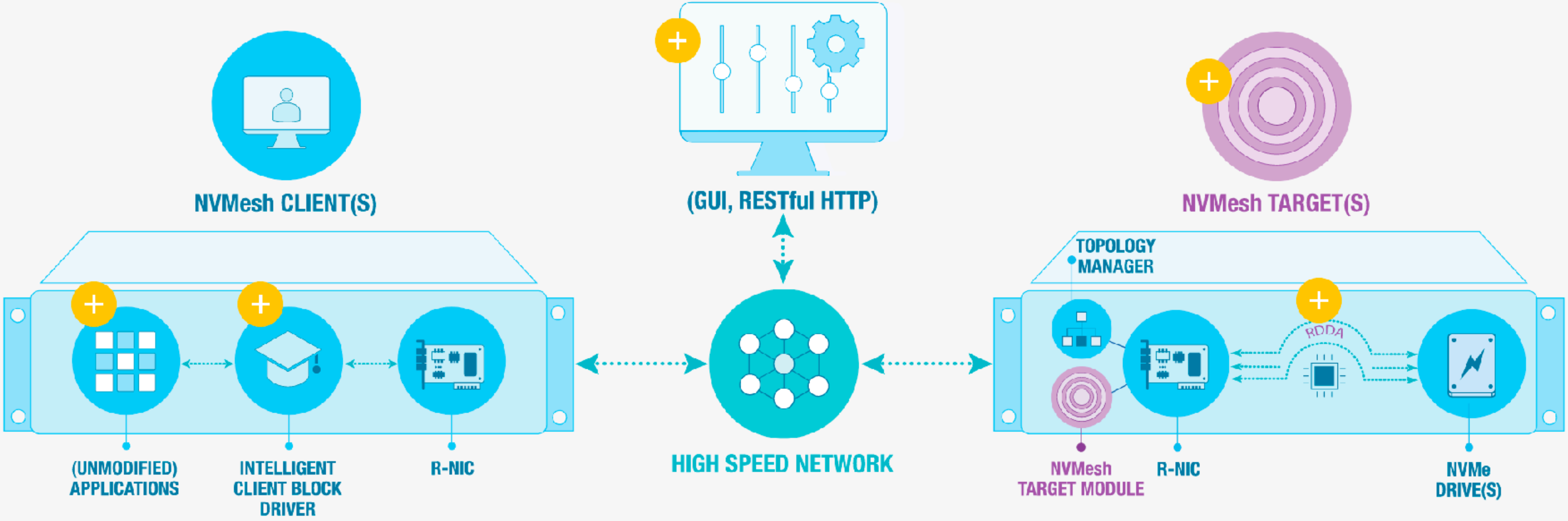
SkyhookDM - Tabular data management in object storage.



IO Server - NVMesh

by **Exceleron**: software defined storage for NVMe's

- patented RDDA - remote direct drive access bypassing CPU
- low-latency scalable block storage solution
- **no open source**



IO Server - NVMesh

Storage Configurator

Example Configuration 10 Quad-Server

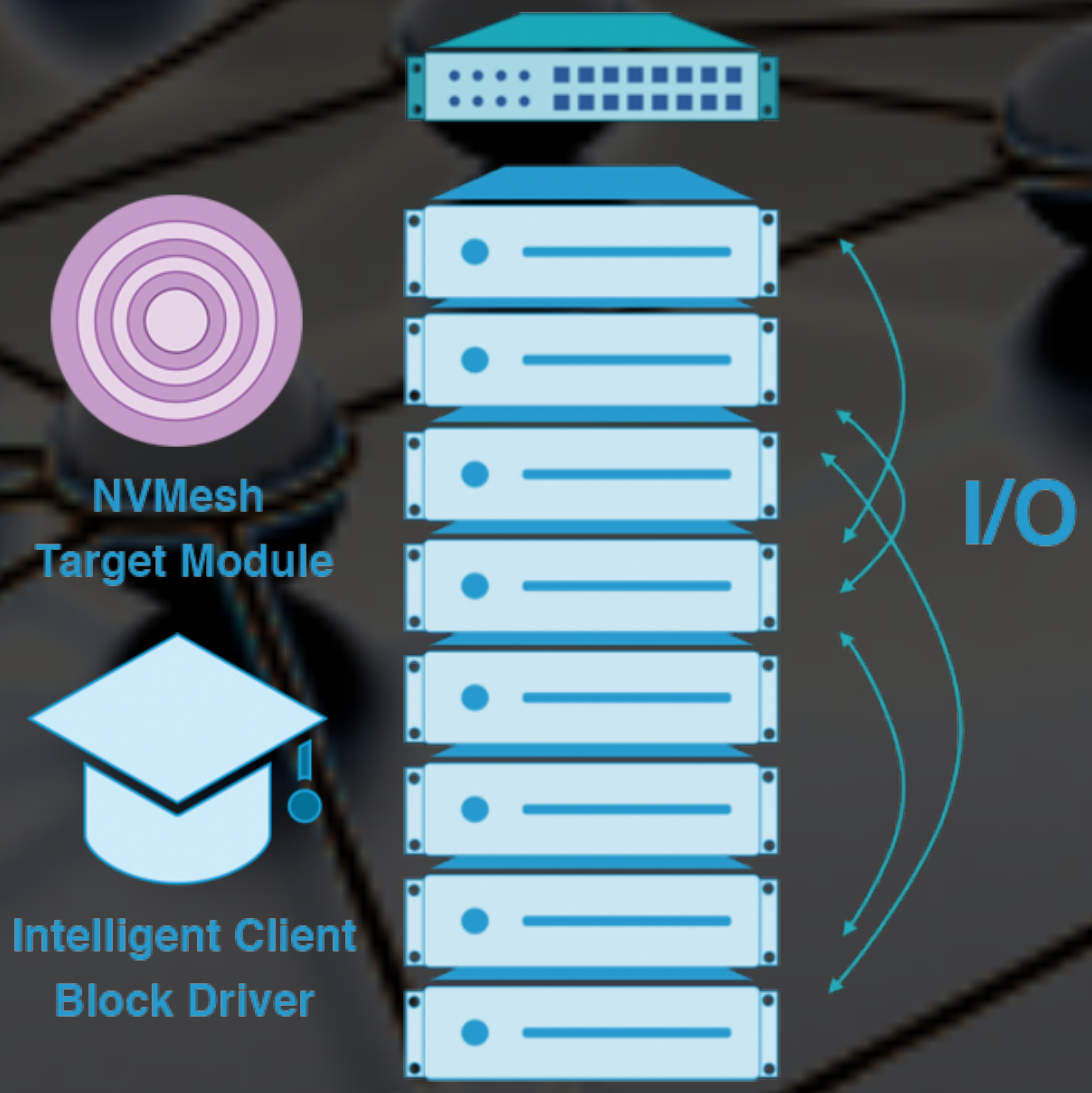
Aggregated Performance

MeshProtect Level	Max 4K IOPS	Max Bandwidth	Latency(µs)		Capacity	
			Read	Write	Base 10	Base 2
MeshProtect 0	200M	785 GB/s	drive+5	25	1.843 TB	1.676 TiB
MeshProtect 10	200M	785 GB/s	drive+5	31	922 TB	838 TiB
MeshProtect EC (6+2)	200M	777 GB/s	drive+5	140	1.382 TB	1.257 TiB
MeshProtect EC (8+2)	200M	777 GB/s	drive+5	140	1.475 TB	1.341 TiB

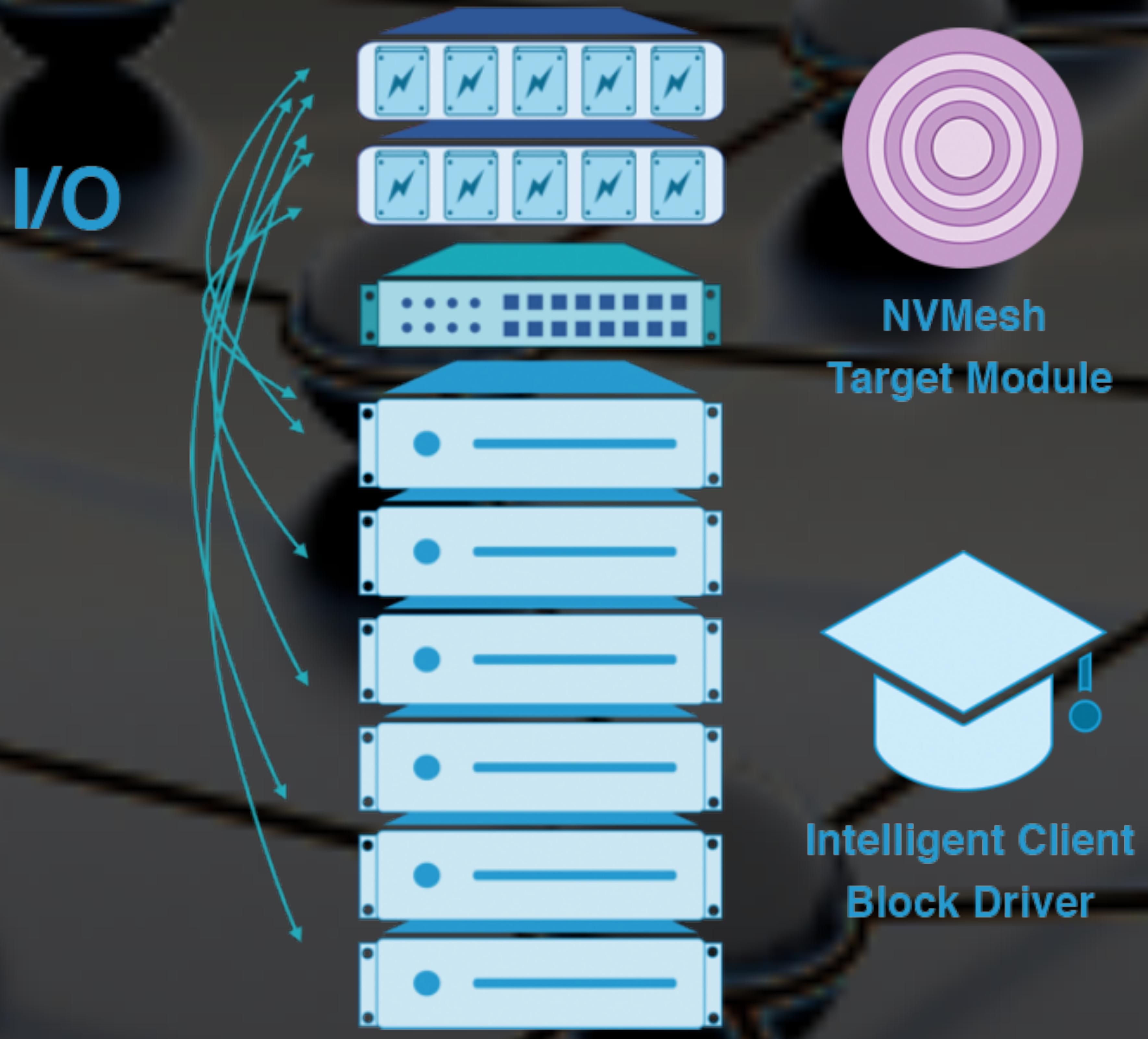
technology like **rbd@CEPH**, client kernel driver,
server is light-weight, erasure coding is done client side

IO Server - NVMesh

hyper converged

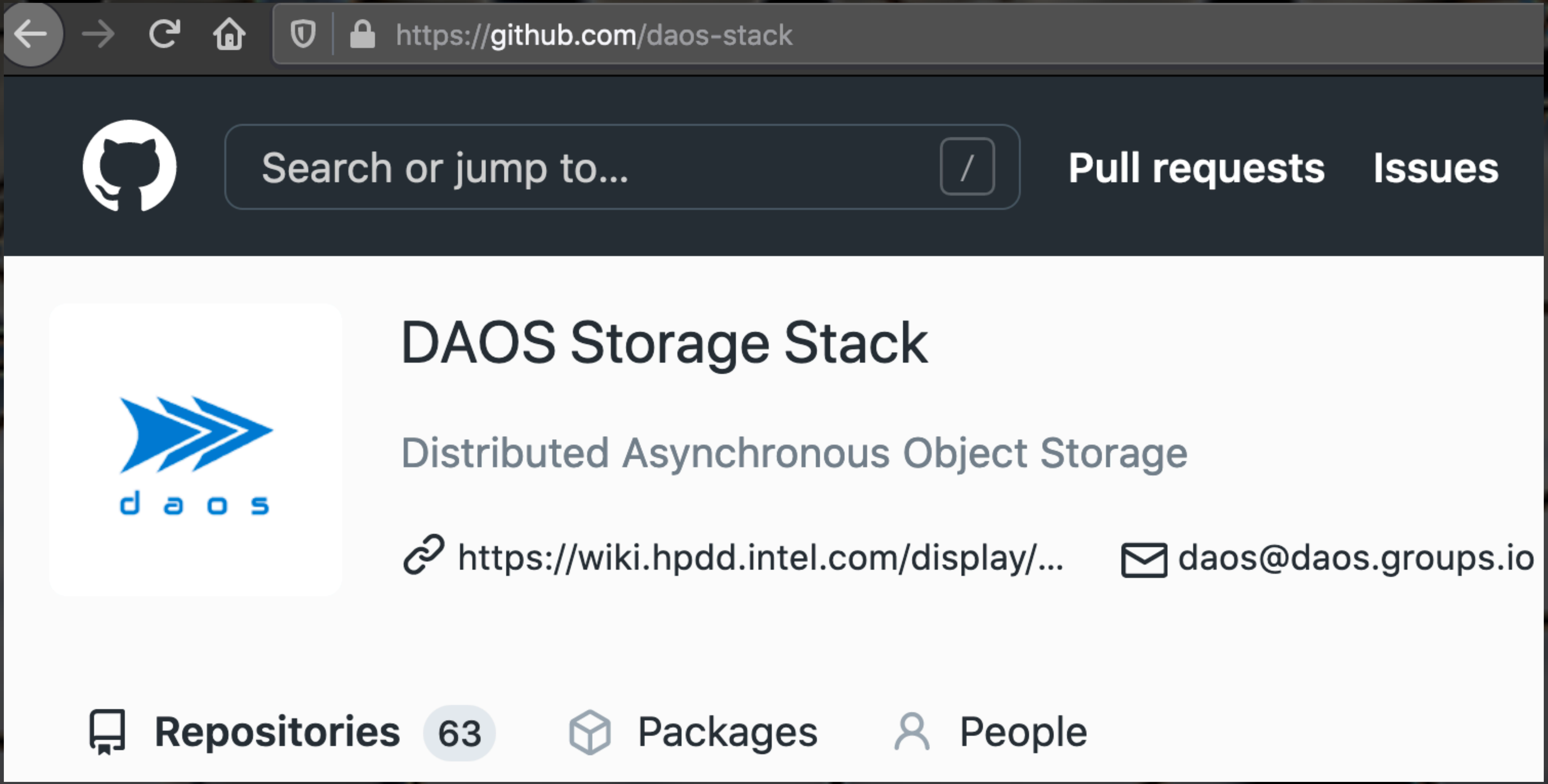


as centralised storage back-end



IO Server - DAOS

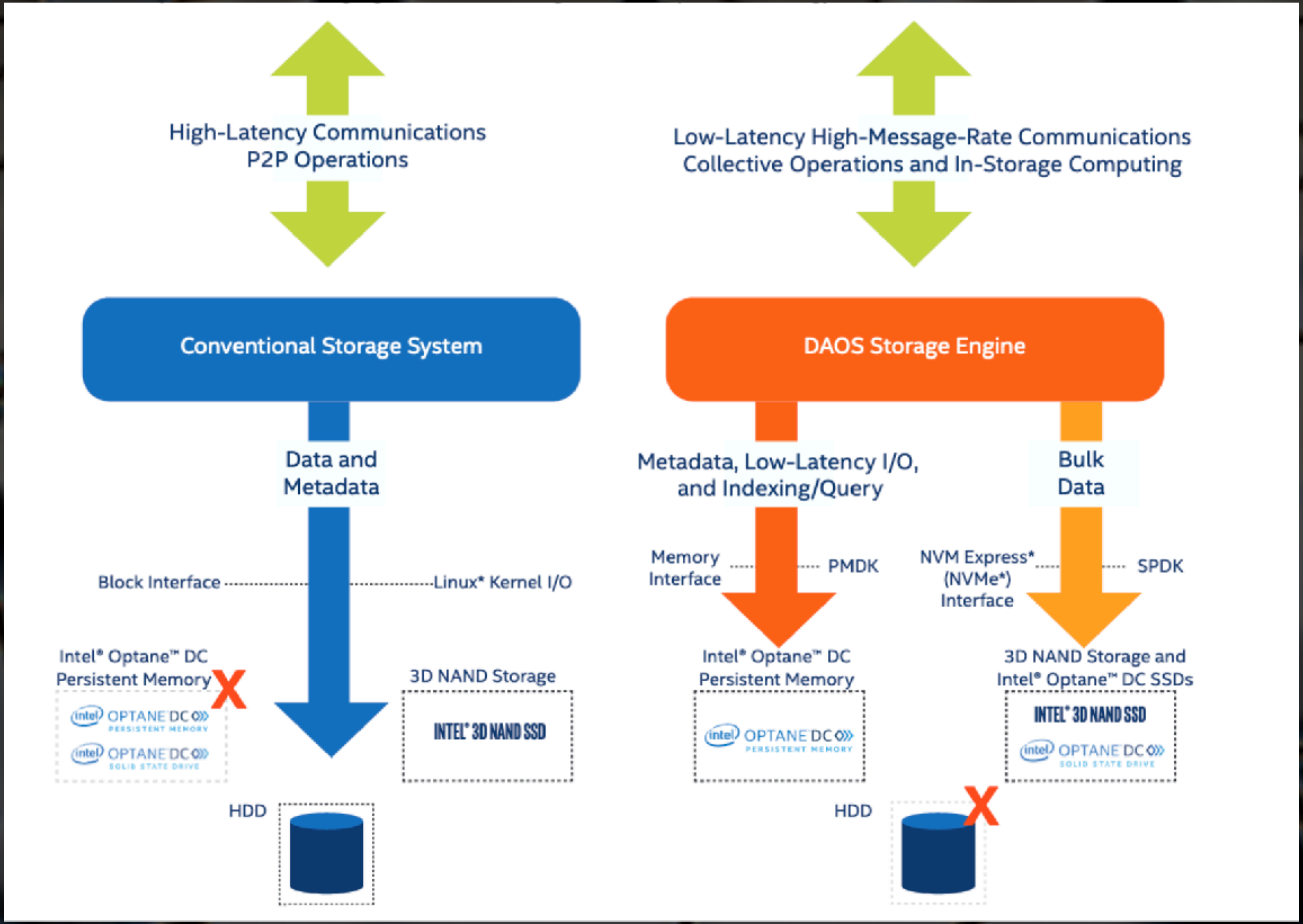
open source distributed asynchronous object storage



software is free … you pay for the hardware

IO Server - DAOS

open source distributed asynchronous object storage

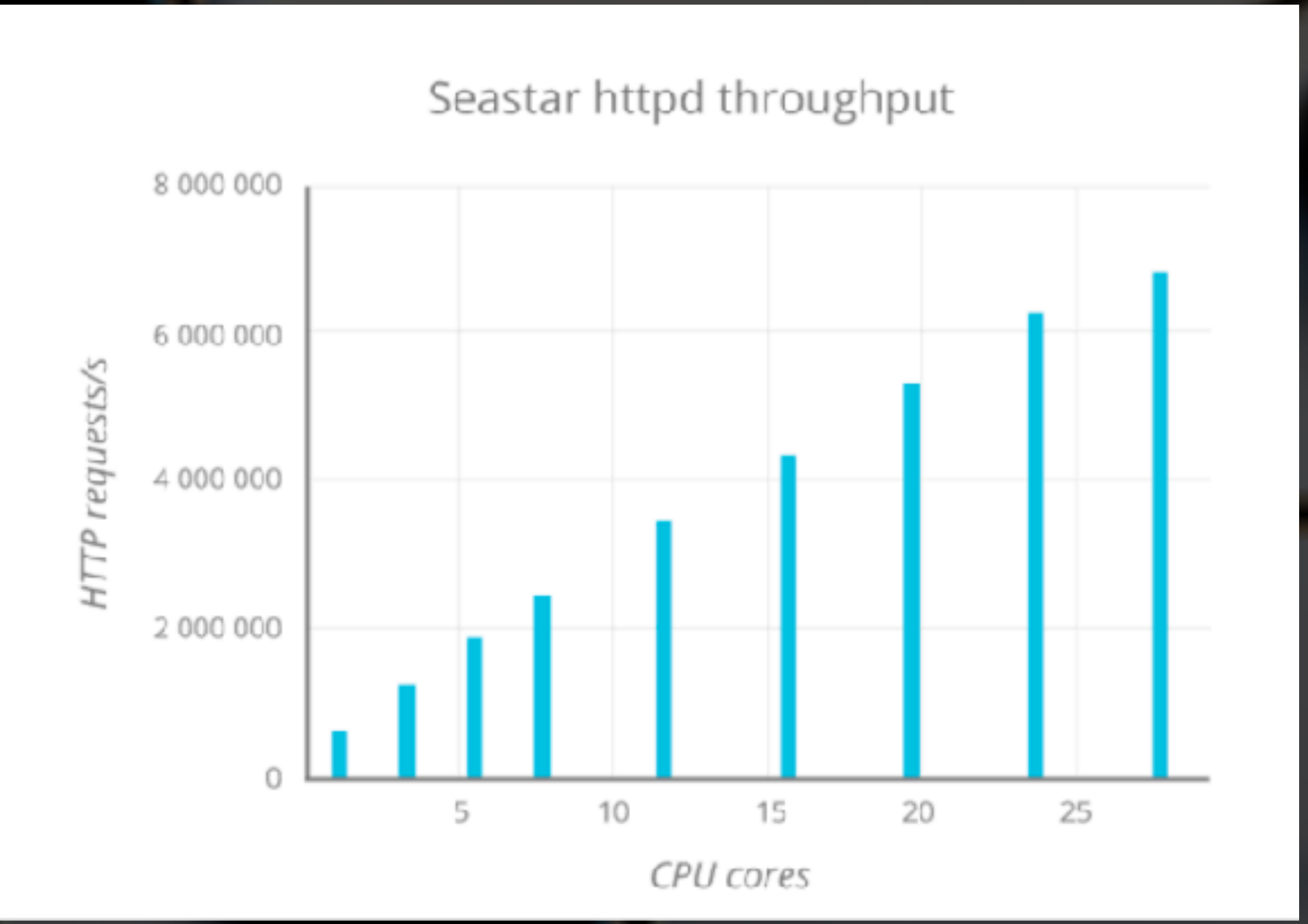


IO Server - DAOS

open source distributed asynchronous object storage

IO Server - CEPH Crimson

- ceph **OSD** re-write with **Seastar** framework http://seastar.io/ (using SPDK) - codename of new OSD: **Crimson**
- longish project - not trivial
- replacement of **BlueStore** e.g. with **PoseidonStore**
  - io_uring, new kernel asynchronous I/O interface to exploit interrupt-driven I/O
  - CPU is bottleneck for several NVMe cards

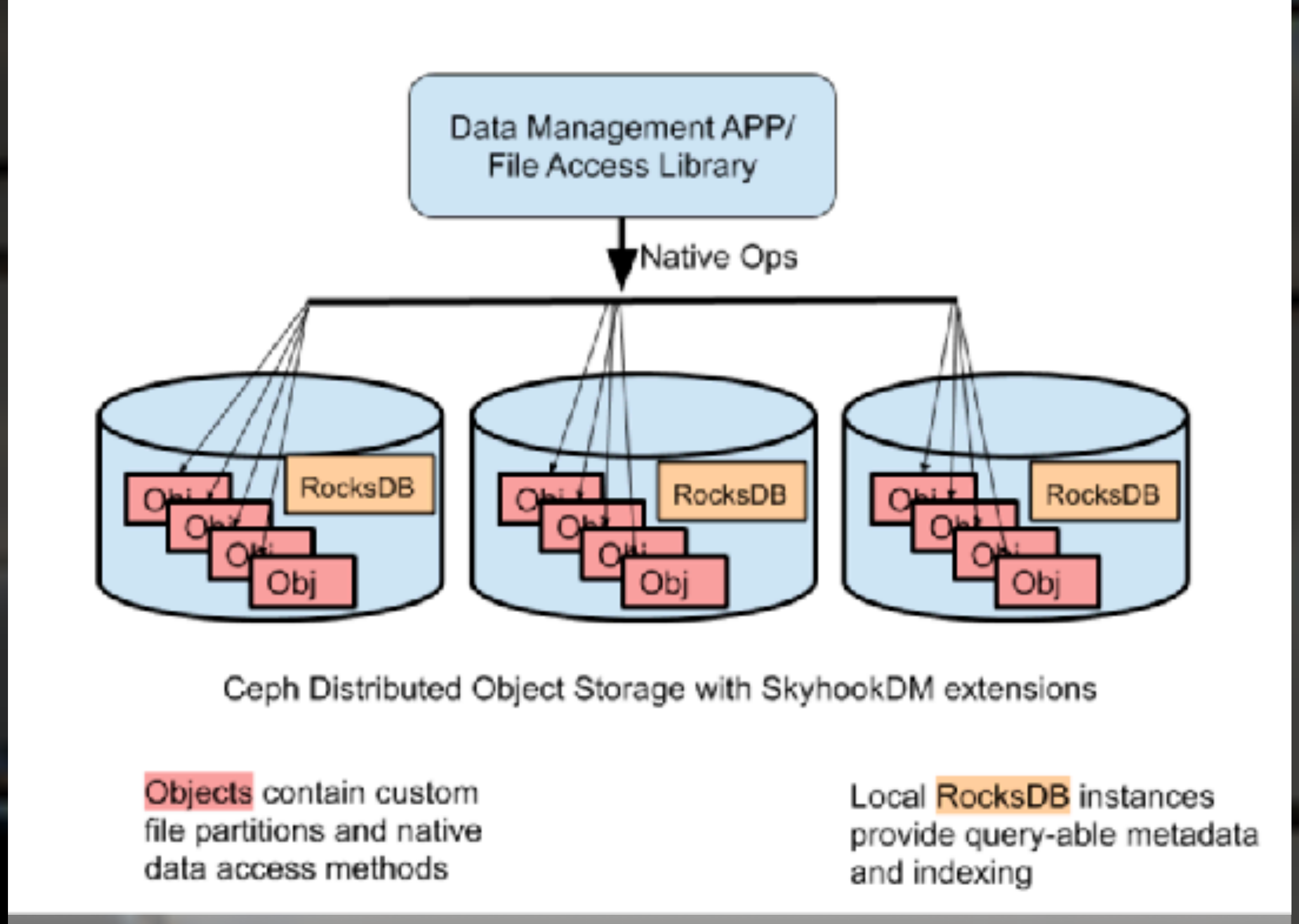


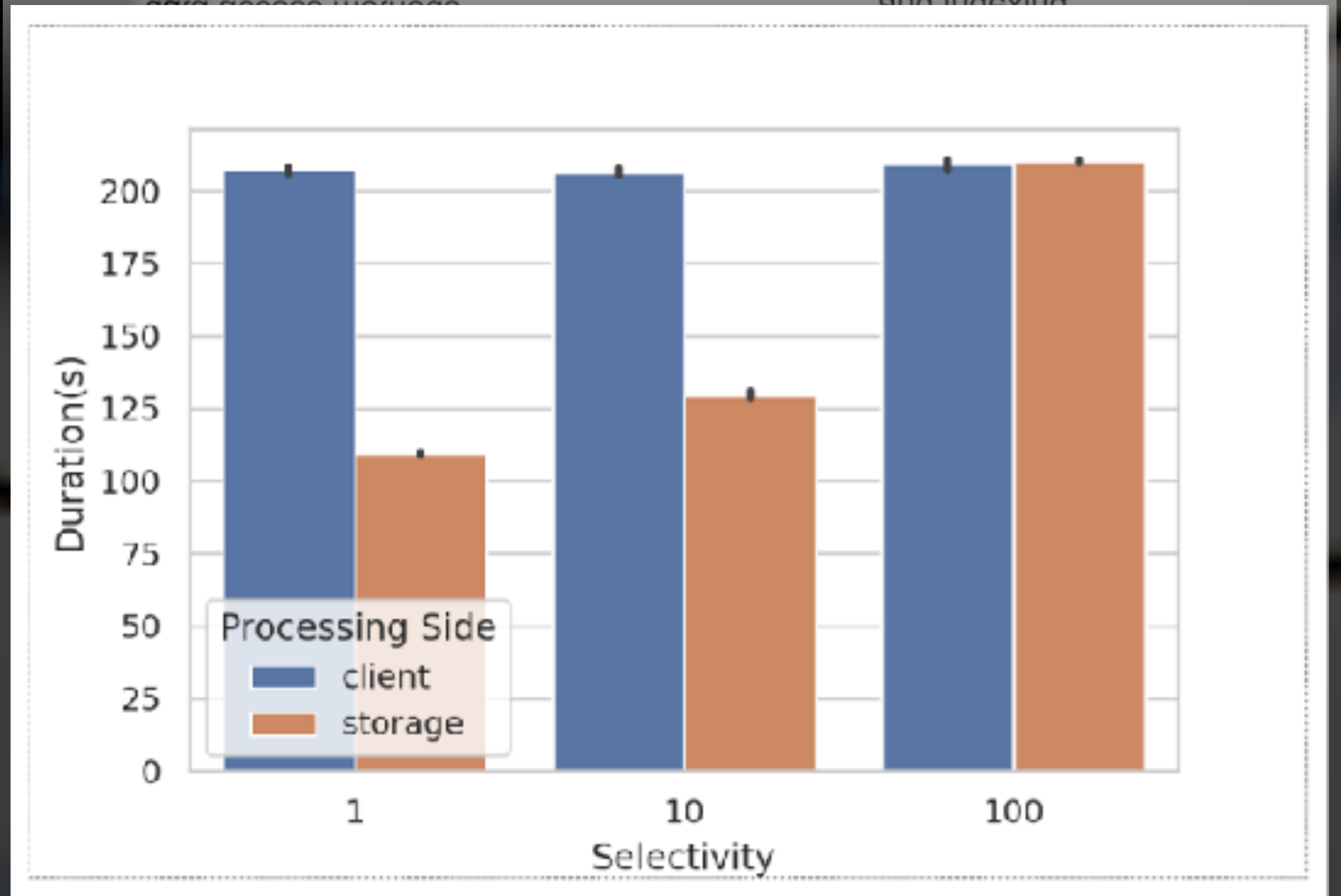
Seastar https server scaling

IO Server - ØNETWORK

SkyhookDM - Tabular data management in object storage.

- Processing inside CEPH object storage (hyper-converged)
- SkyhookDM supports row-based processing via [Google Flatbuffers format](#) and col-based processing via [Apache Arrow](#) fast in-memory serialization formats





Time spent in querying 1%, 10%, and 100% data with query processing on client and storage side

https://medium.com/getpopper/reproducible-experiments-and-benchmarks-on-skyhookdm-ceph-using-popper-64c42d47a65a

IO Server - ØNETWORK

SkyhookDM - Tabular data management in object storage.

- benefits from object locality
  - advantage is less obvious with erasure encoded objects (remote access)
- benefits clearly in selective processing
- Questions
  - can you have LHC frameworks running on your storage system (memory, software updates etc.) ?
  - is the performance benefit worth it?

Summary

future **IO** requirements are less problematic than **CPU** in HEP
-**100 GE + parallel IO** with HDDs provide required order of magnitude improvements essentially today

SSD/NVMe useful as ad-hoc storage at CERN

- exploring IOPS of flash storage requires modernised server async IO approach *(when will we have kernel 5.1 at CERN?)*
  - temporary solution: physics applications should avoid too small IO requests (100GE = 40k x 256kb)

Open Source Storage Software has become **mainstream**. There are fantastic developments ongoing to provide ultra low latency storage systems, which can revolutionise data formats and processing. For now they are not cost competitive for an organisation like CERN.

you can join the EOS virtual workshop: eos.cern.ch