STORAGE DEVELOPER CONFERENCE

SDC 21

BY Developers FOR Developers

Virtual Conference
September 28-29, 2021

SkyhookDM: An Arrow-Native Storage System

Jayjeet Chakraborty, Carlos Maltzahn
Centre for Research in Open Source Software
UC Santa Cruz

The Broader Problem

- CPU is the new bottleneck with modern high speed storage and network devices like NVMe and Infiniband networks

- Client-side computation of data and reading from efficient storage formats like Parquet, ORC exhausts the clients CPUs

- Scalability and Latency is severely hampered.

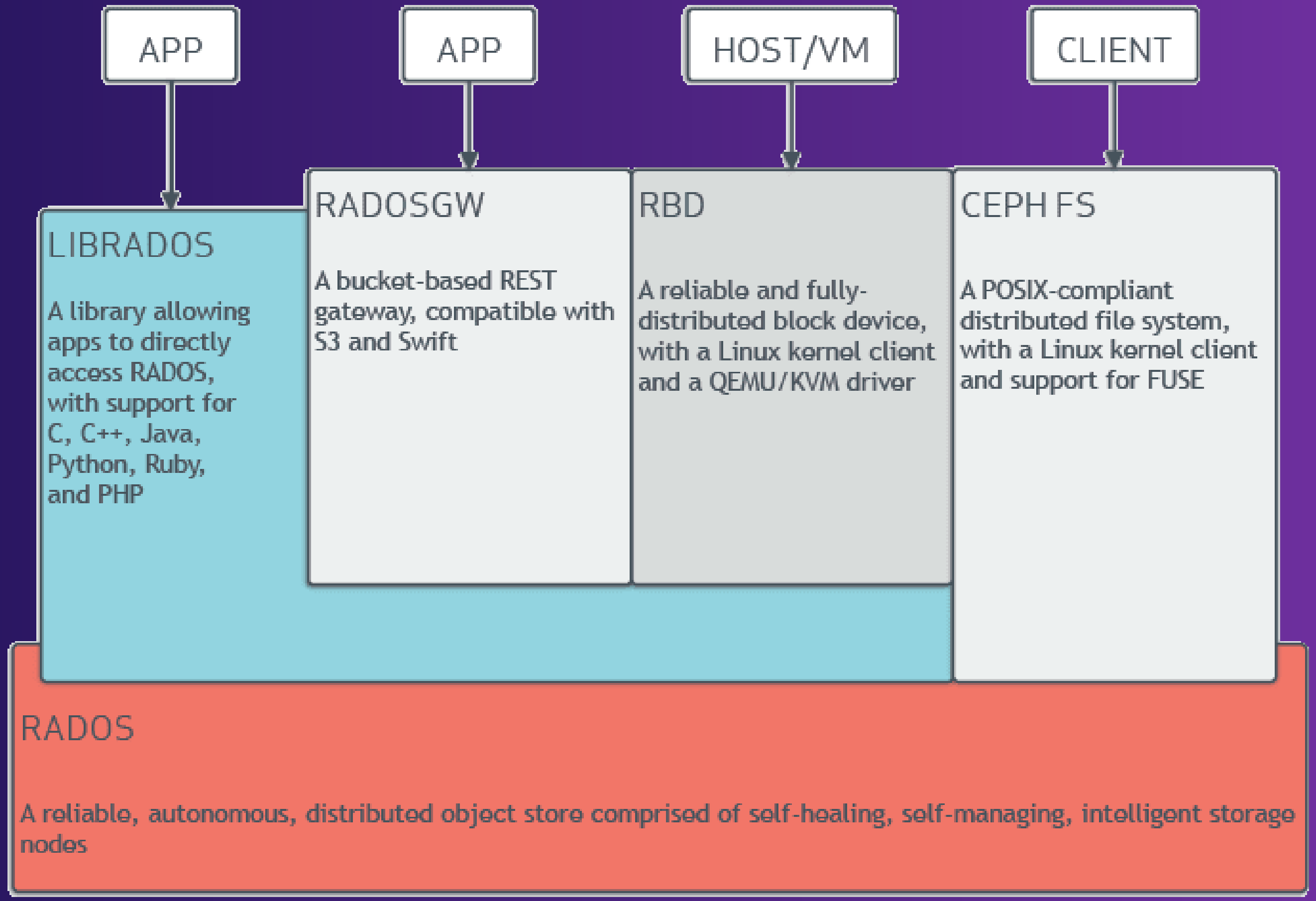
Computational Storage as a Solution

- Offload computation from the client to the storage layer as much as possible
- Utilize the idle CPUs of storage systems for increased processing rates and faster queries
- Results in lesser data movement, memory copying, and network traffic

Ceph

Introduction

- Provides 3 types of storage interface: File, Object, Block
- No central point of failure. Uses CRUSH maps that contains Object - OSD mapping
- Extensible Object storage layer via the Ceph Object Classes SDK



APP

APP

HOST/VM

CLIENT

LIBRADOS

A library allowing apps to directly access RADOS, with support for C, C++, Java, Python, Ruby, and PHP

RADOSGW

A bucket-based REST gateway, compatible with S3 and Swift

RBD

A reliable and fully-distributed block device, with a Linux kernel client and a QEMU/KVM driver

CEPH FS

A POSIX-compliant distributed file system, with a Linux kernel client and support for FUSE

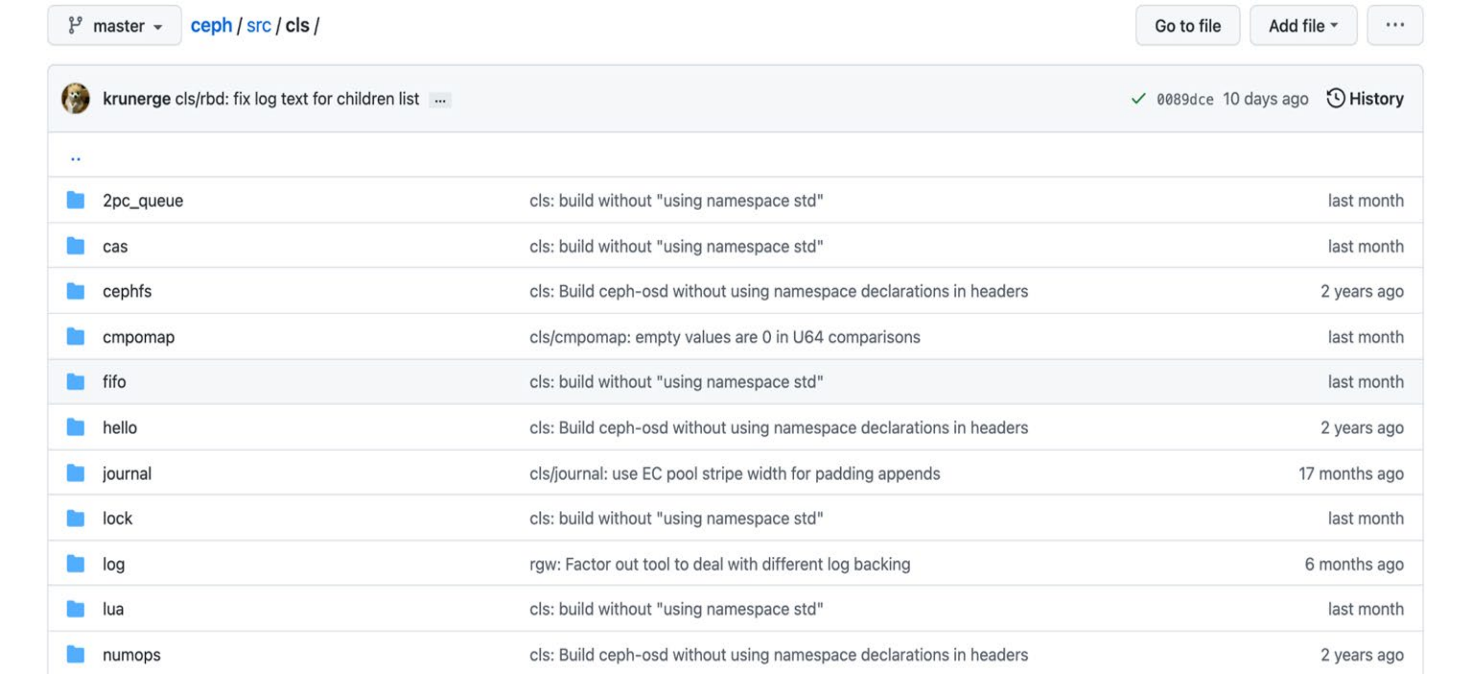
RADOS

A reliable, autonomous, distributed object store comprised of self-healing, self-managing, intelligent storage nodes

Object Class Mechanism

- Utilizing Ceph’s object class mechanism (“cls”)
  - Object storage extension mechanism
  - Present in ceph/src/cls
- Used by several Ceph internals
  - CephFS, RGW, RBD

Object Classes in Ceph

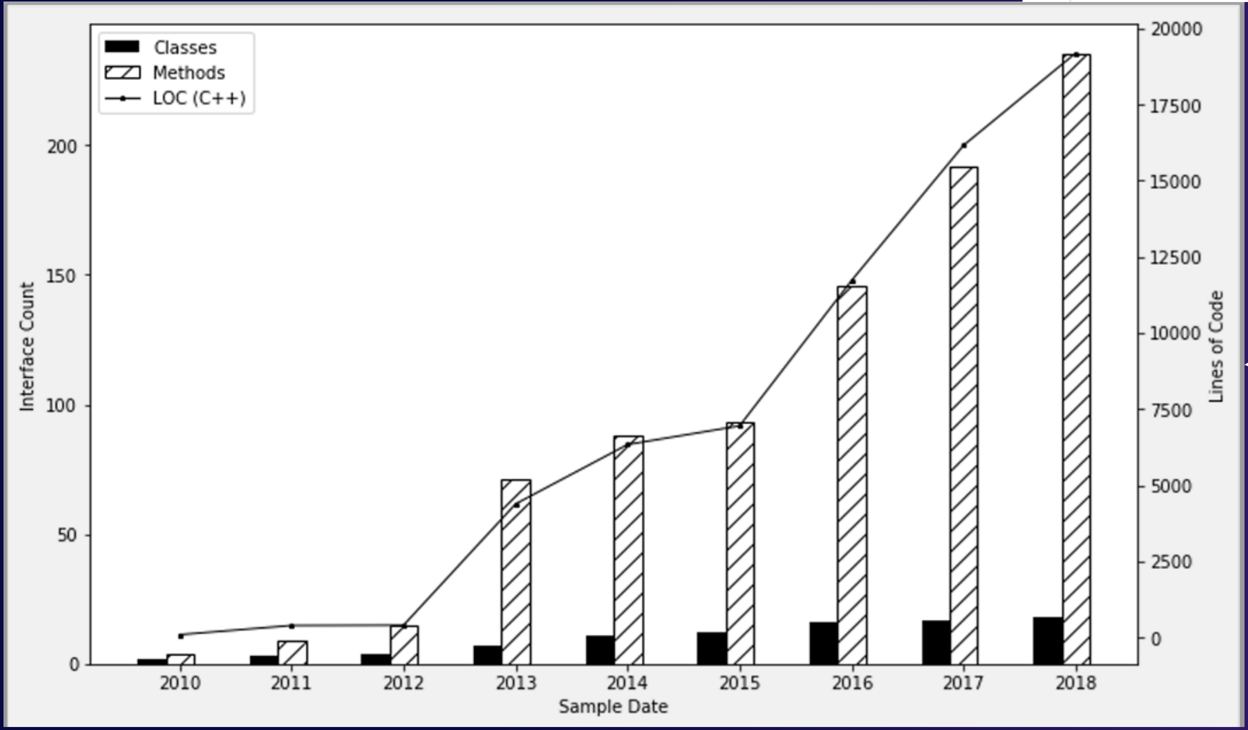


ceph / src / cls /

krunerge cls/rbd: fix log text for children list ... ✓ 0089dce 10 days ago History

..

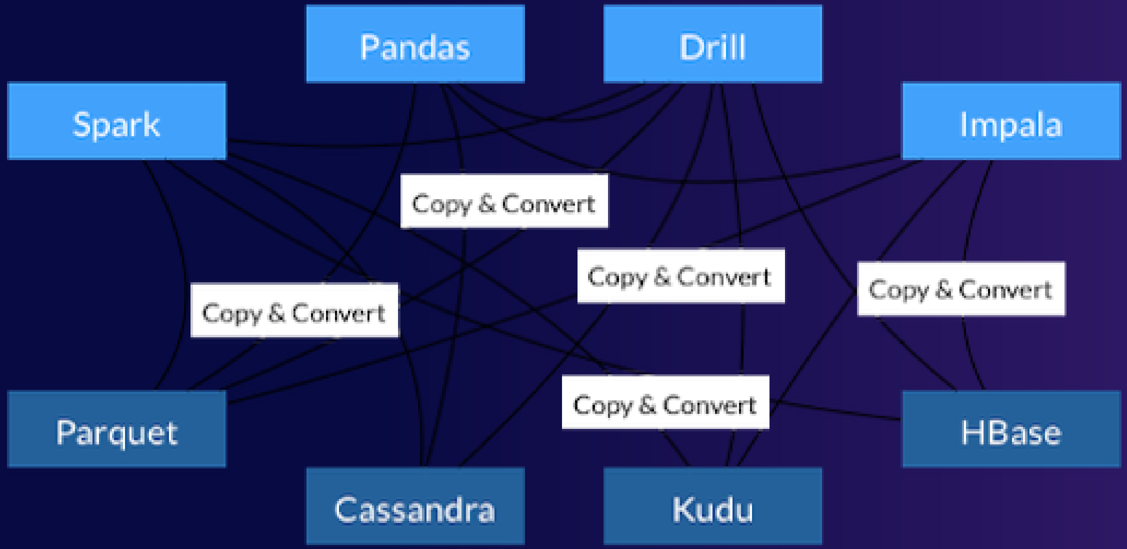
2pc_queue	cls: build without "using namespace std"	last month
cas	cls: build without "using namespace std"	last month
cephfs	cls: Build ceph-osd without using namespace declarations in headers	2 years ago
cmpomap	cls/cmpomap: empty values are 0 in U64 comparisons	last month
fifo	cls: build without "using namespace std"	last month
hello	cls: Build ceph-osd without using namespace declarations in headers	2 years ago
journal	cls/journal: use EC pool stripe width for padding appends	17 months ago
lock	cls: build without "using namespace std"	last month
log	rgw: Factor out tool to deal with different log backing	6 months ago
lua	cls: build without "using namespace std"	last month
numops	cls: Build ceph-osd without using namespace declarations in headers	2 years ago

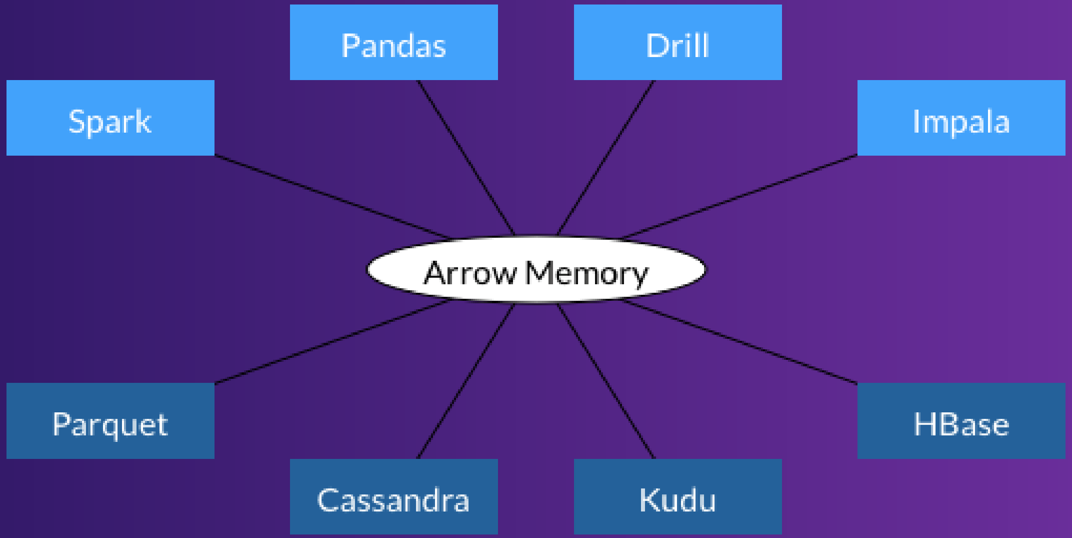


Growth of Object Classes in Ceph

Apache Arrow

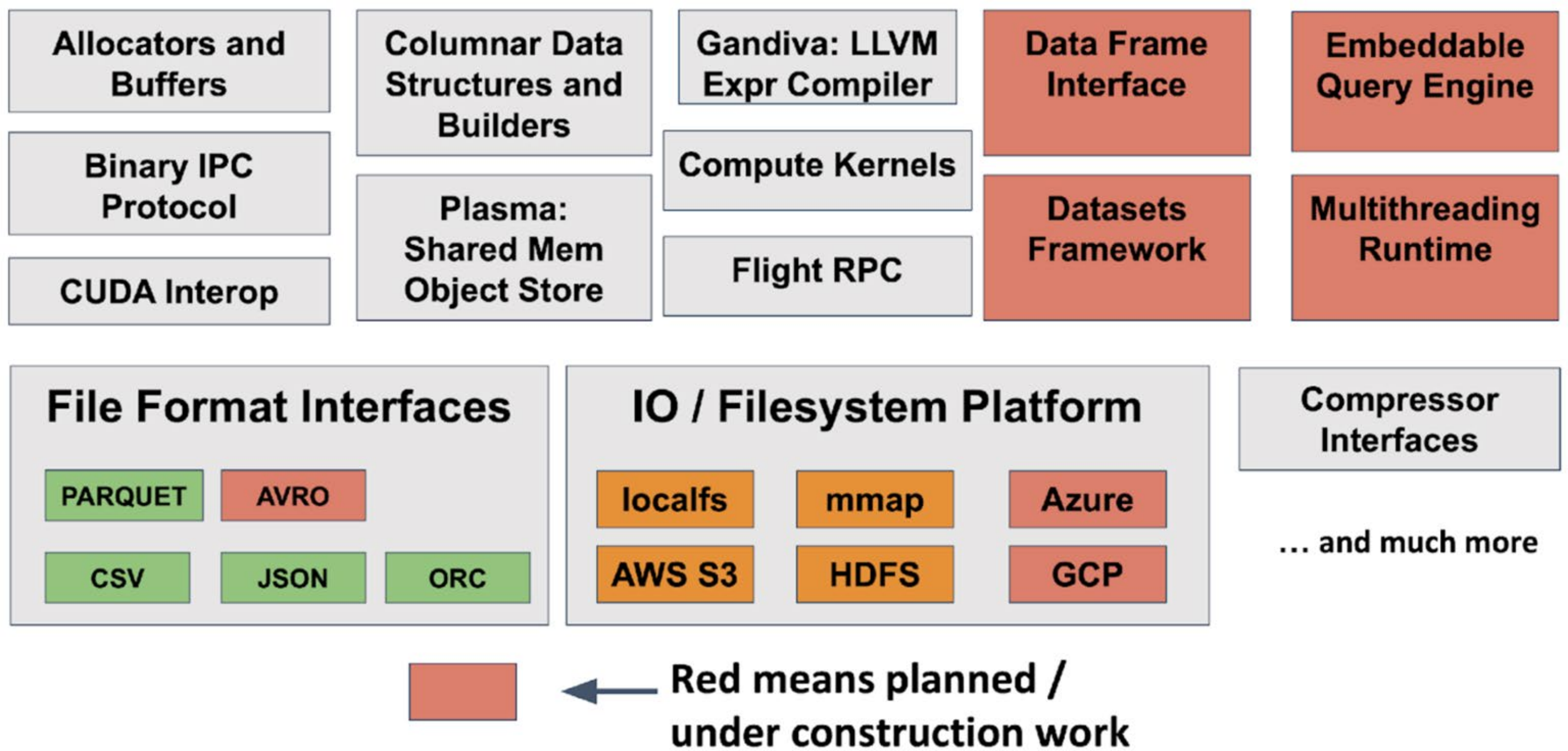
- Language-independent columnar memory format for flat and hierarchical data, organised for efficient analytic operations on modern hardware





What else ?

- Rich collection of pluggable components for building data processing systems



Allocators and Buffers

Binary IPC Protocol

CUDA Interop

Columnar Data Structures and Builders

Plasma: Shared Mem Object Store

Gandiva: LLVM Expr Compiler

Compute Kernels

Flight RPC

Data Frame Interface

Datasets Framework

Embeddable Query Engine

Multithreading Runtime

File Format Interfaces

PARQUET AVRO

CSV JSON ORC

IO / Filesystem Platform

localfs mmap Azure

AWS S3 HDFS GCP

Compressor Interfaces

… and much more

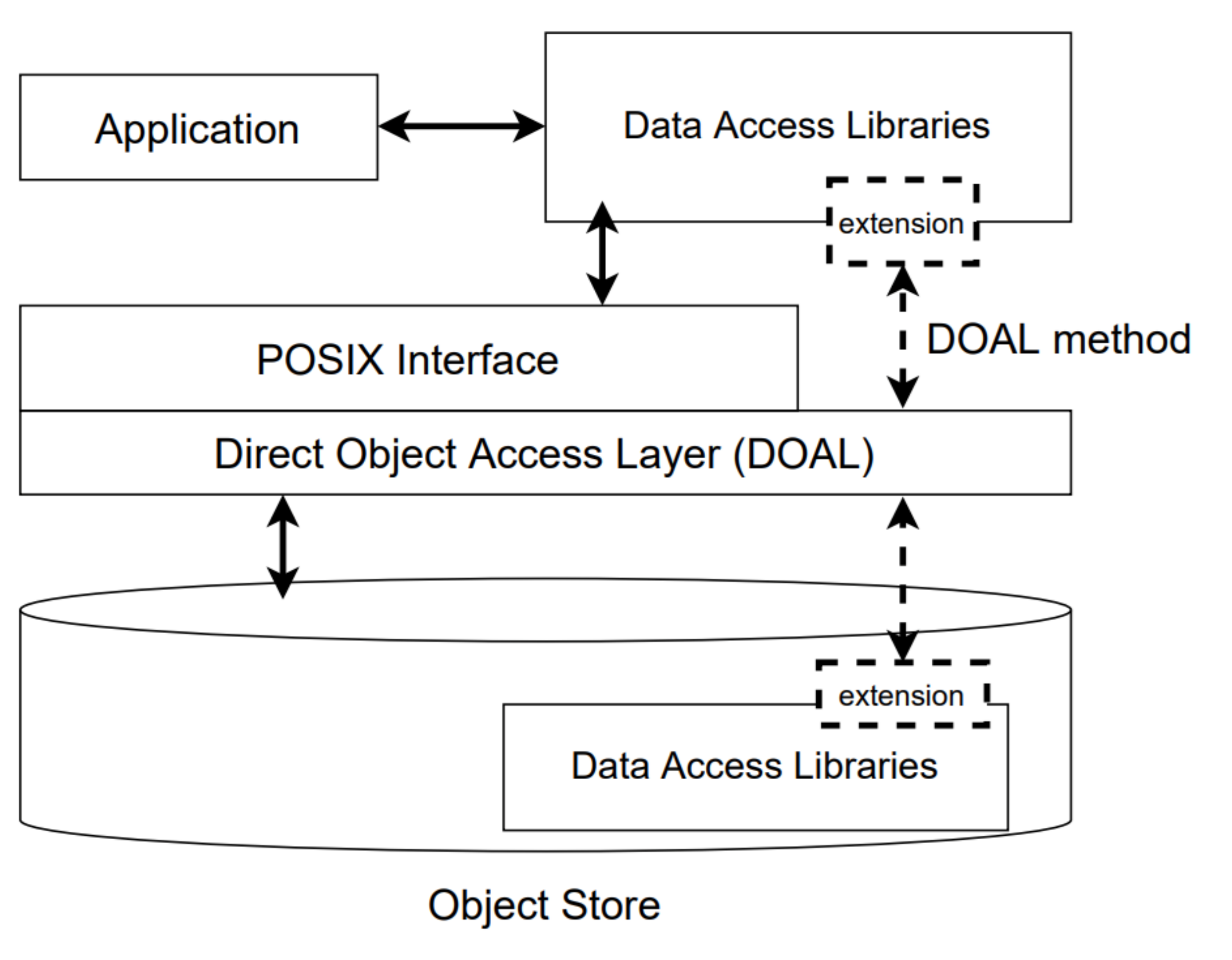
🟥 ← Red means planned / under construction work

Design Paradigm

- Extend client and storage layers of programmable storage systems with data access libraries

- Embed a FS shim inside storage nodes to have file-like view over objects

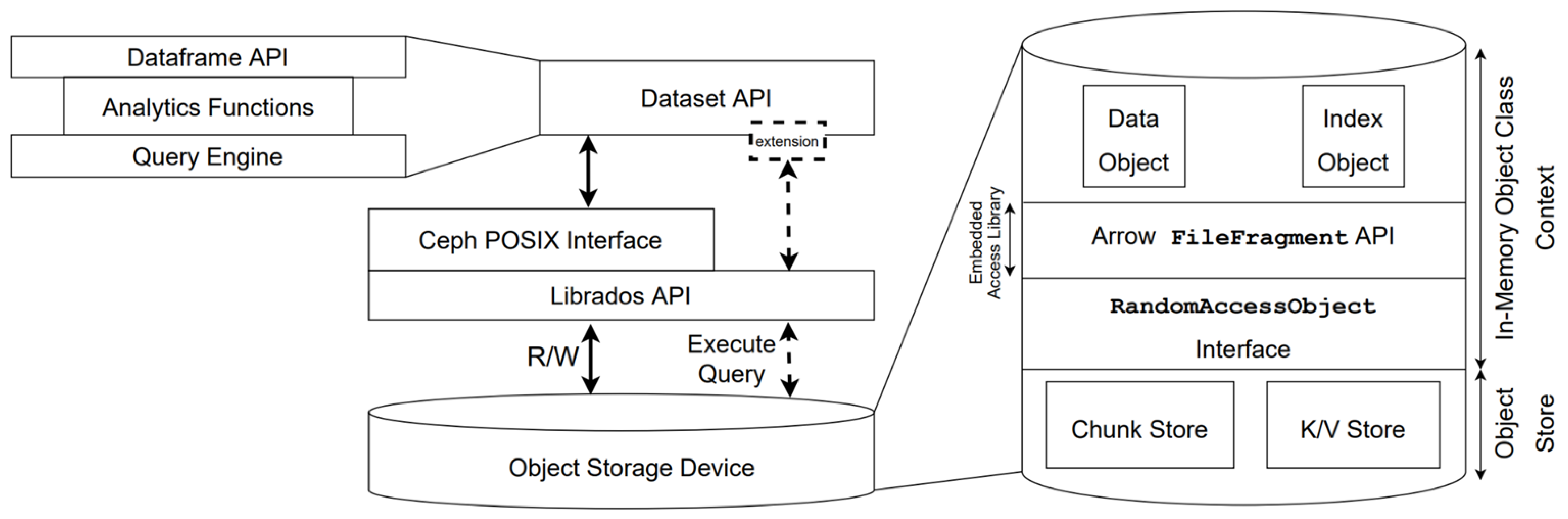
- Allow direct interaction with objects in an object store while bypassing the filesystem layer utilising FS metadata



Architecture

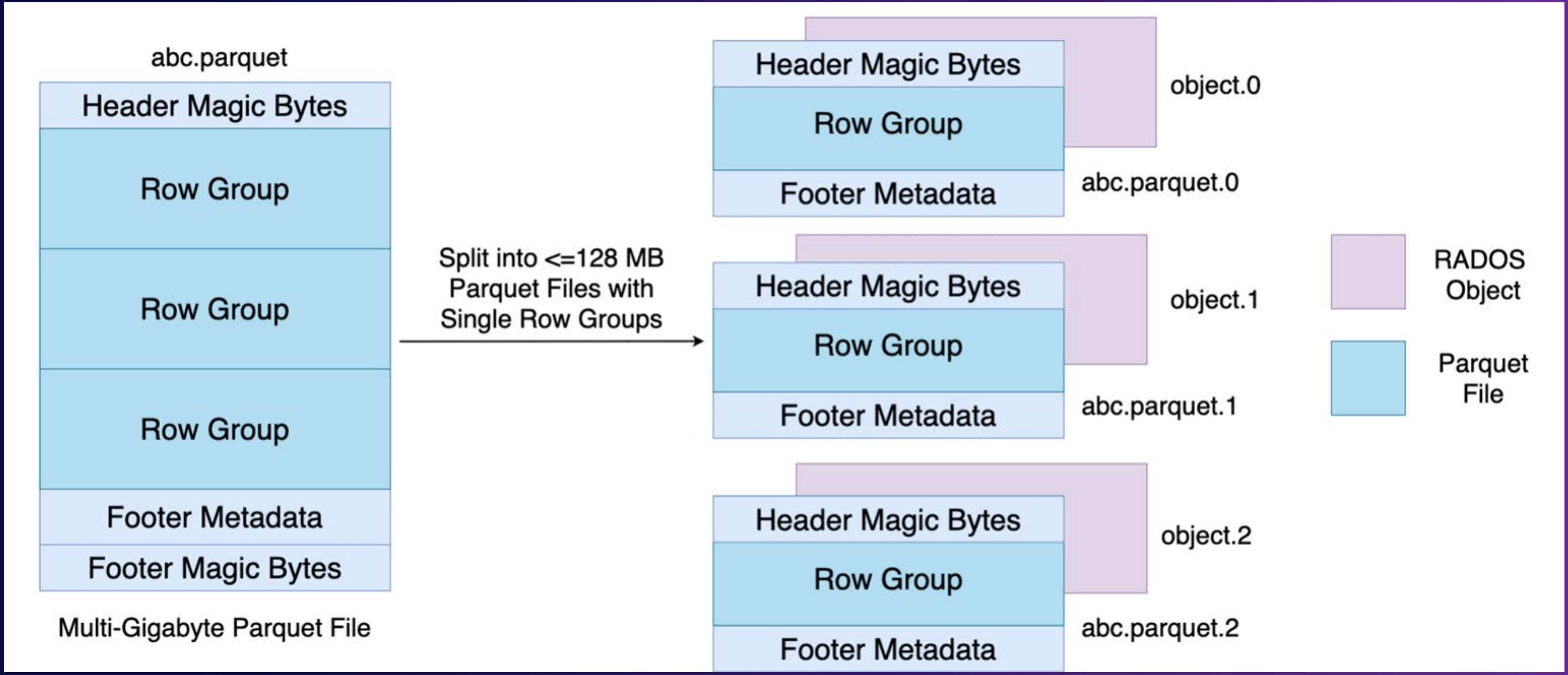
- Arrow data access libraries embedded inside Ceph OSDs to allow scanning data fragments in the Ceph storage layer

- Extend Arrow Dataset API with `SkyhookFileFormat` to expose the offload capability



File-Layout Design

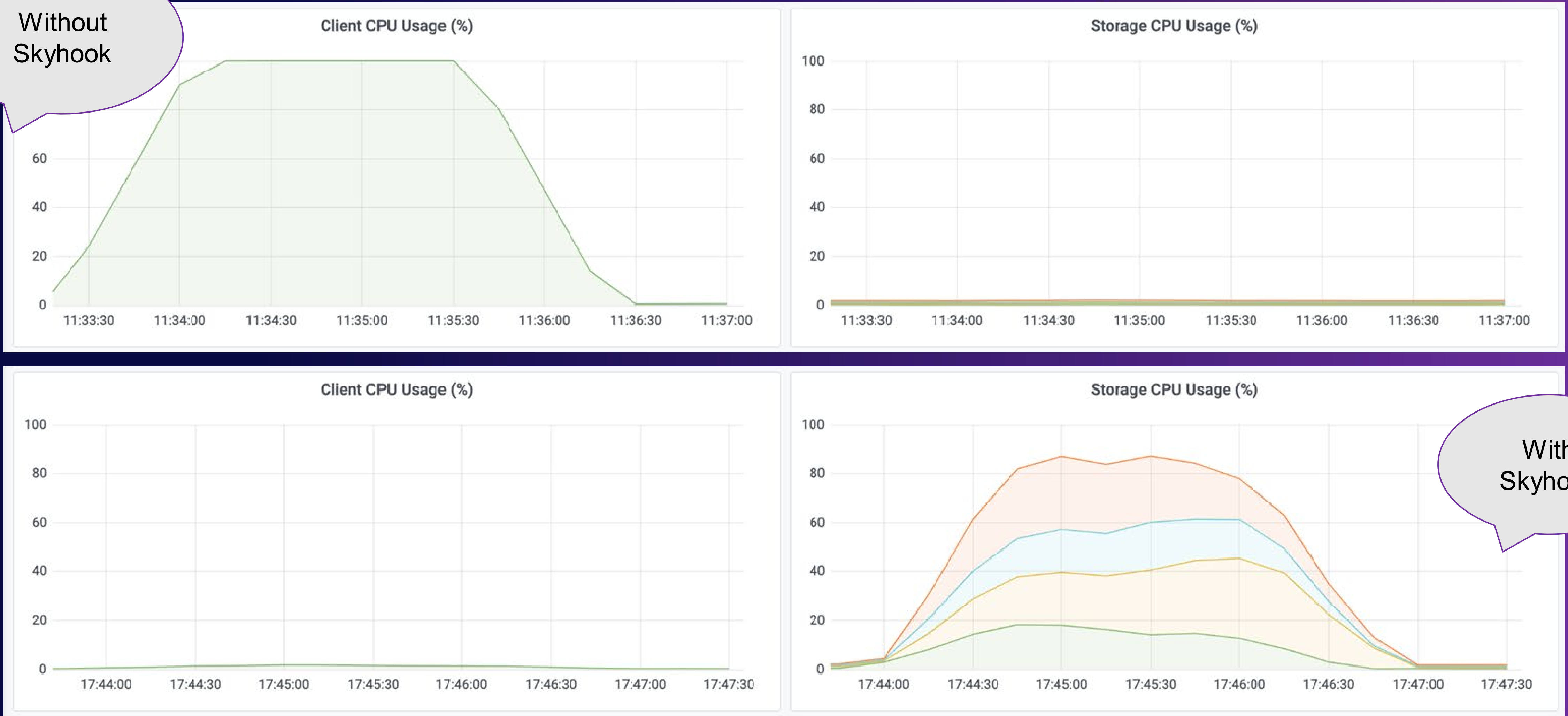
- 16MB is the preferred file size in SkyhookDM as found out from several experiments with different file sizes.
- Files larger than 16MB are splitted into smaller files of ~16MB and each file is stored in a single RADOS object.
- Due to Arrow Dataset API being the data access library, a wide range of file formats like IPC, Parquet, CSV are supported out of the box.


abc.parquet
Header Magic Bytes
Row Group
Row Group
Row Group
Footer Metadata
Footer Magic Bytes
Multi-Gigabyte Parquet File
Split into <=128 MB Parquet Files with Single Row Groups
Header Magic Bytes
Row Group
Footer Metadata
object.0
abc.parquet.0
Header Magic Bytes
Row Group
Footer Metadata
object.1
abc.parquet.1
Header Magic Bytes
Row Group
Footer Metadata
object.2
abc.parquet.2
RADOS Object
Parquet File

Results

Offloaded CPU usage

Without Skyhook



Client CPU Usage (%)

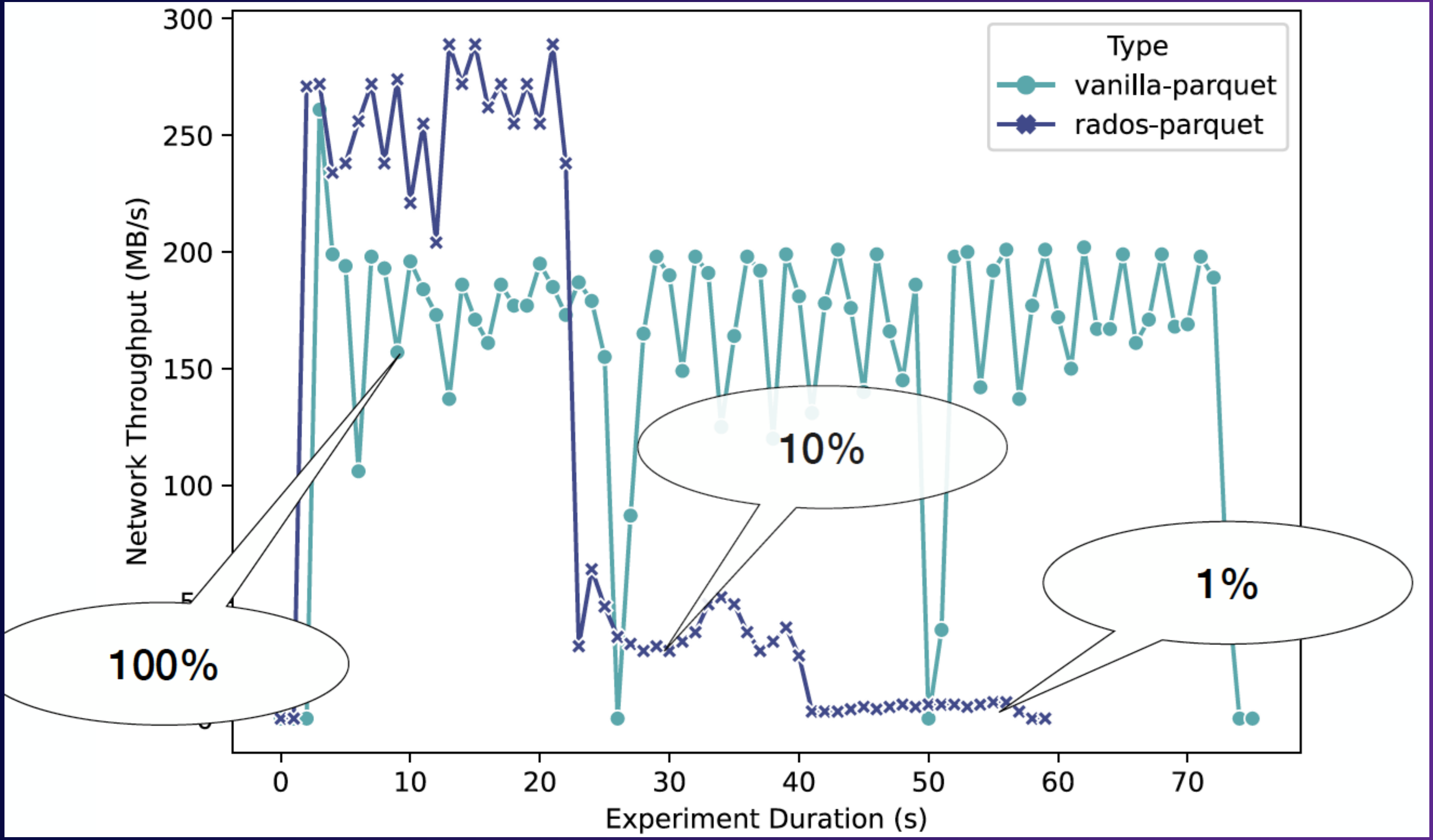
Storage CPU Usage (%)

Client CPU Usage (%)

Storage CPU Usage (%)

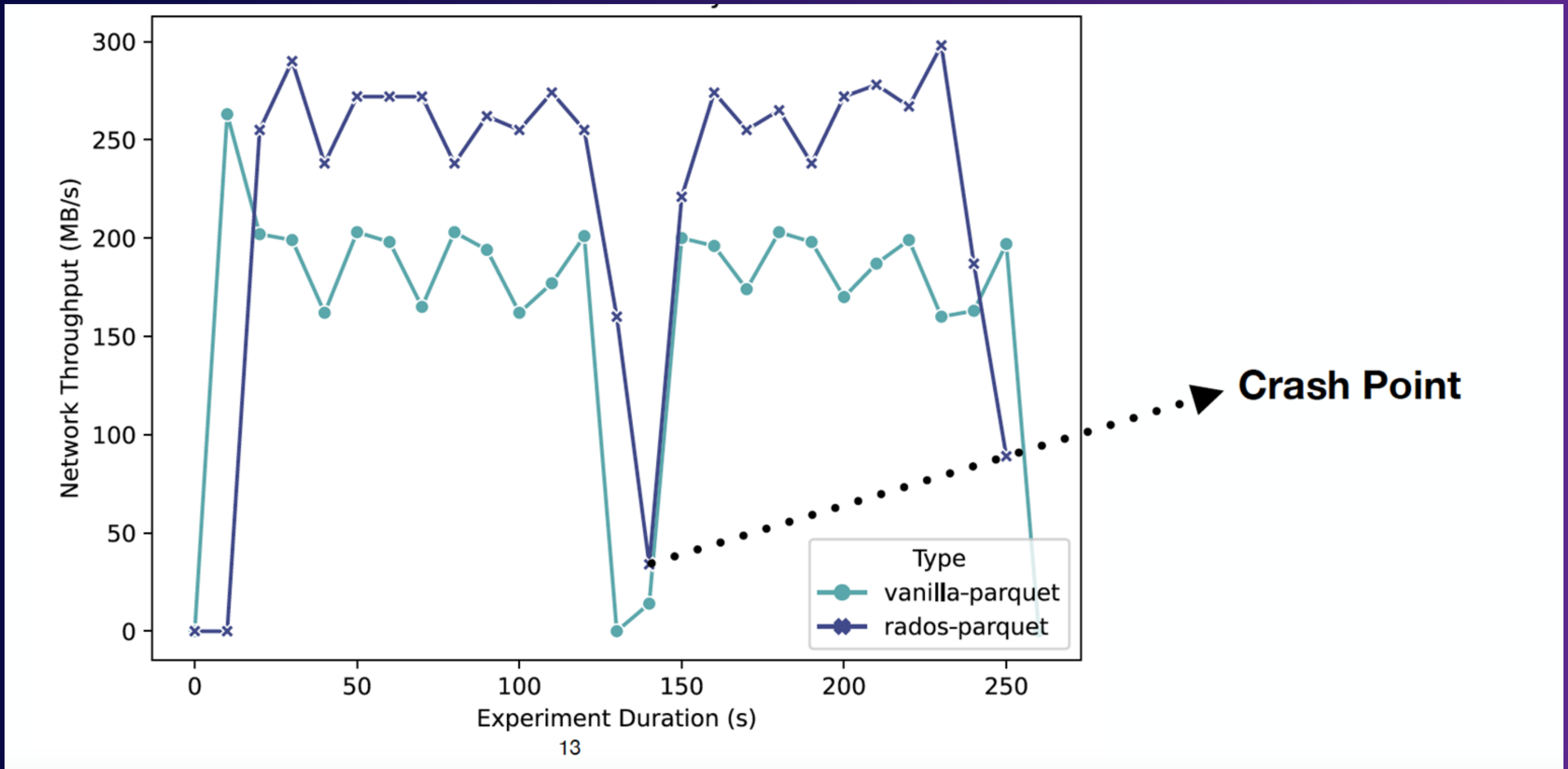
With Skyhook

Reduced Wastage of Network Bandwidth



Automatic Failure Recovery

Since, compute is colocated with storage nodes, the failure recovery and consistency semantics of the storage system apply naturally to the query processing layer



Please take a moment to rate this session

Thank You !