Facebook's Tectonic Filesystem: Efficiency from Exascale

Satadru Pan[1], Theano Stavrinos[1,2], Yunqiao Zhang[1], Atul Sikaria[1], Pavel Zakharov[1], Abhinav Sharma[1], Shiva Shankar P[1], Mike Shuey[1], Richard Wareing[1], Monika Gangapuram[1], Guanglei Cao[1], Christian Preseau[1], Pratap Singh[1], Kestutis Patiejunas[1], JR Tipton[1], Ethan Katz-Bassett[3], and Wyatt Lloyd[2]

[1]*Facebook, Inc.*, [2]*Princeton University*, [3]*Columbia University*

Exabyte-Scale Storage Use Cases at FB

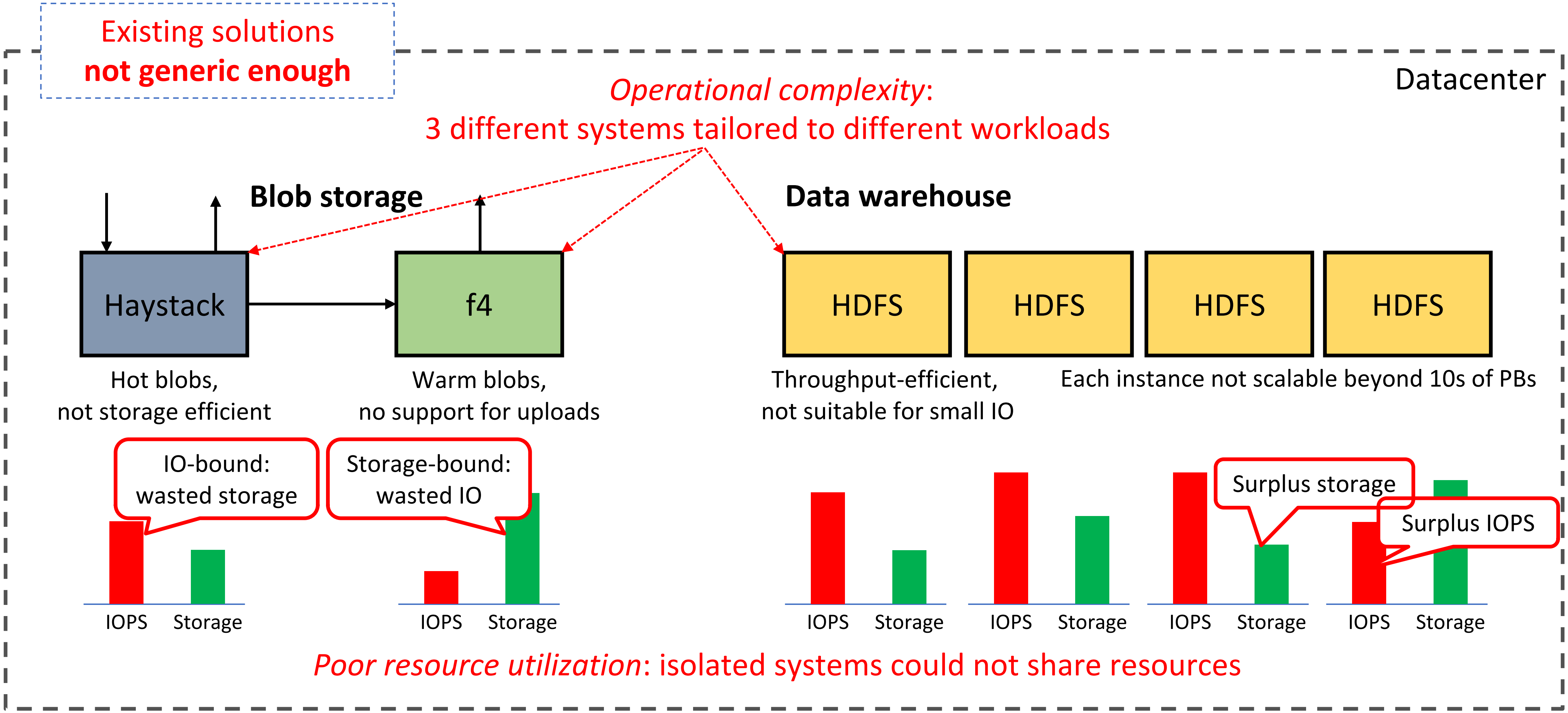
Blob storage

- Photos and videos in Facebook, Messenger attachments
- Exabytes of data
- Several KBs to several MBs in size
- Latency sensitive

Data warehouse

- Hive tables for data analytics, machine learning
- Exabytes of data
- Reads are order of multiple MBs, writes are 10s of MBs
- Throughput sensitive

Storage Infrastructure Before Tectonic



Existing solutions **not generic enough**

Datacenter

Operational complexity: 3 different systems tailored to different workloads

Blob storage

Haystack

Hot blobs, not storage efficient

IO-bound: wasted storage

f4

Warm blobs, no support for uploads

Storage-bound: wasted IO

Data warehouse

HDFS

HDFS

HDFS

HDFS

Throughput-efficient, not suitable for small IO

Each instance not scalable beyond 10s of PBs

Surplus storage

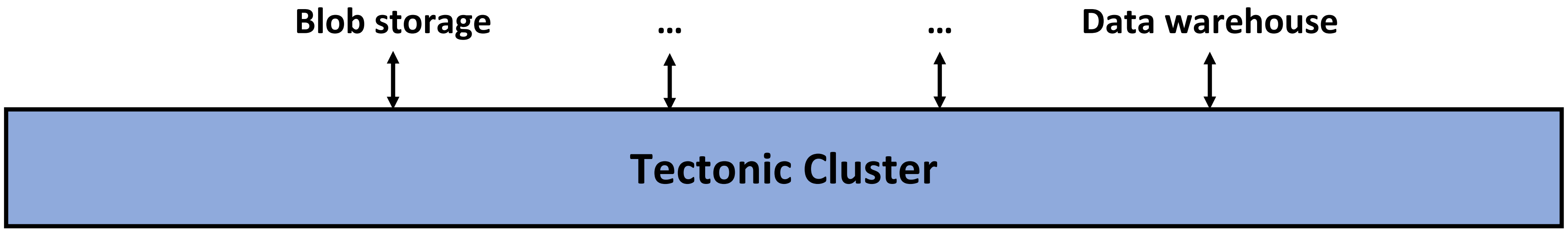
Surplus IOPS

Poor resource utilization: isolated systems could not share resources

Tectonic Overview

Datacenter

Simpler Operations:
Single system to reason about, generic enough to handle all use cases



Better utilization: no stranding of resources

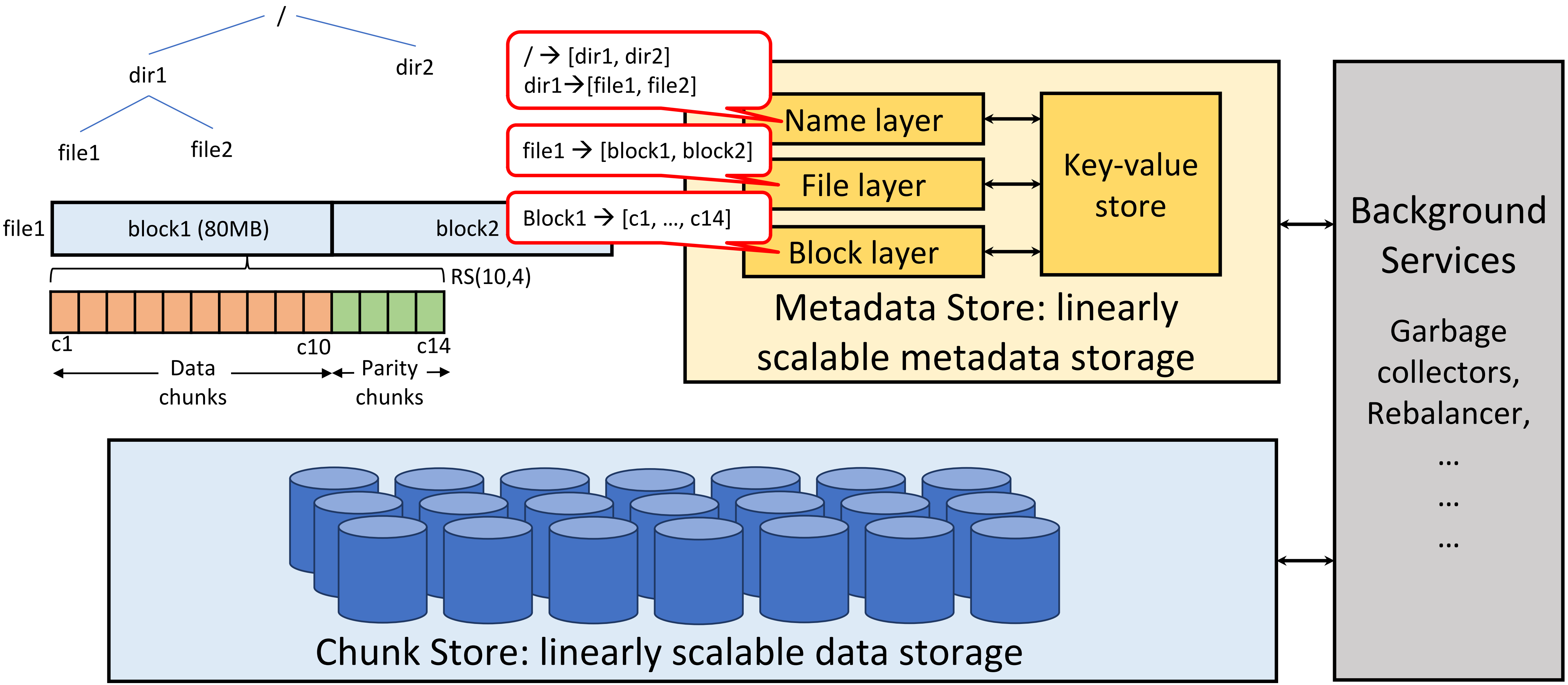
IOPS
Storage

Scalability: support exabyte-scale clusters

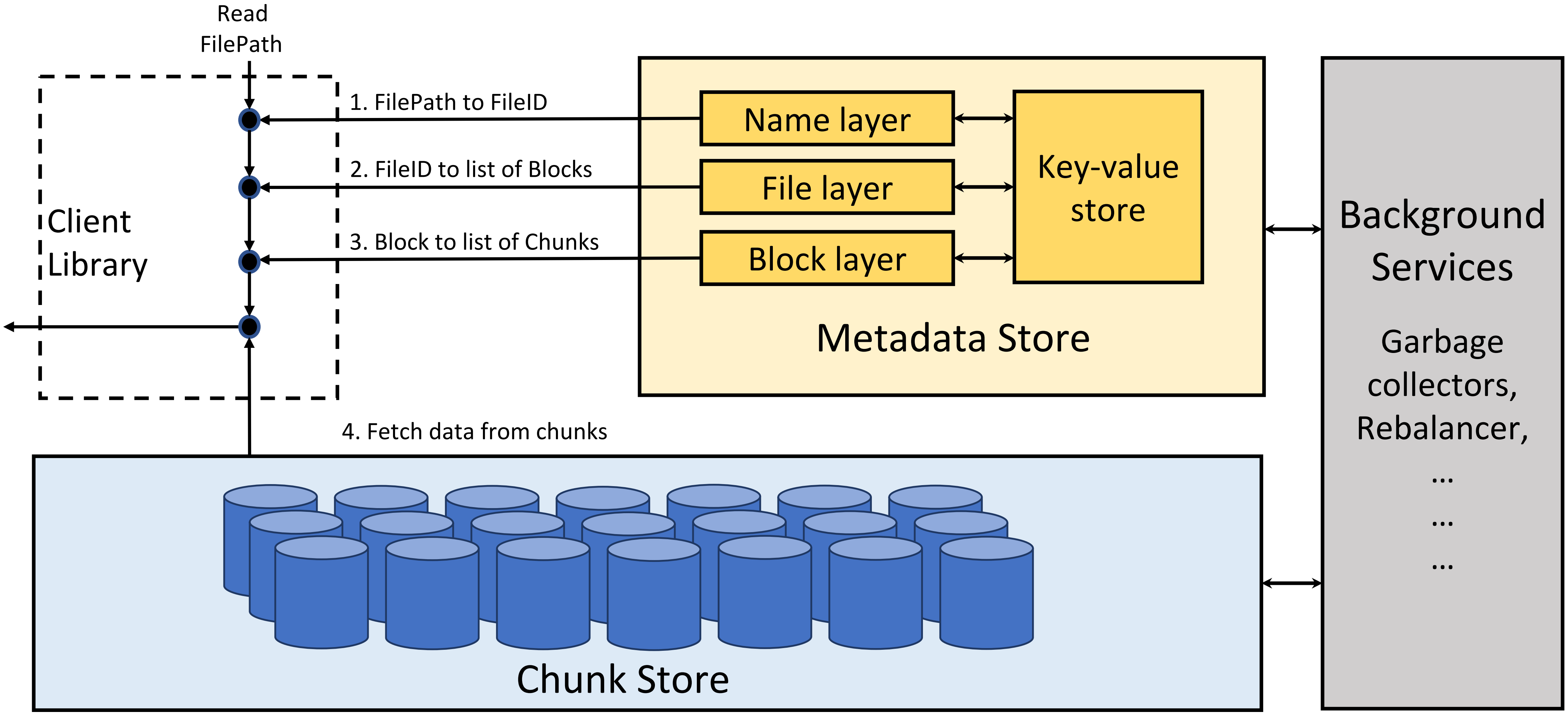
Multitenancy: isolate tenants and share resources

Performance: match performance of specialized systems

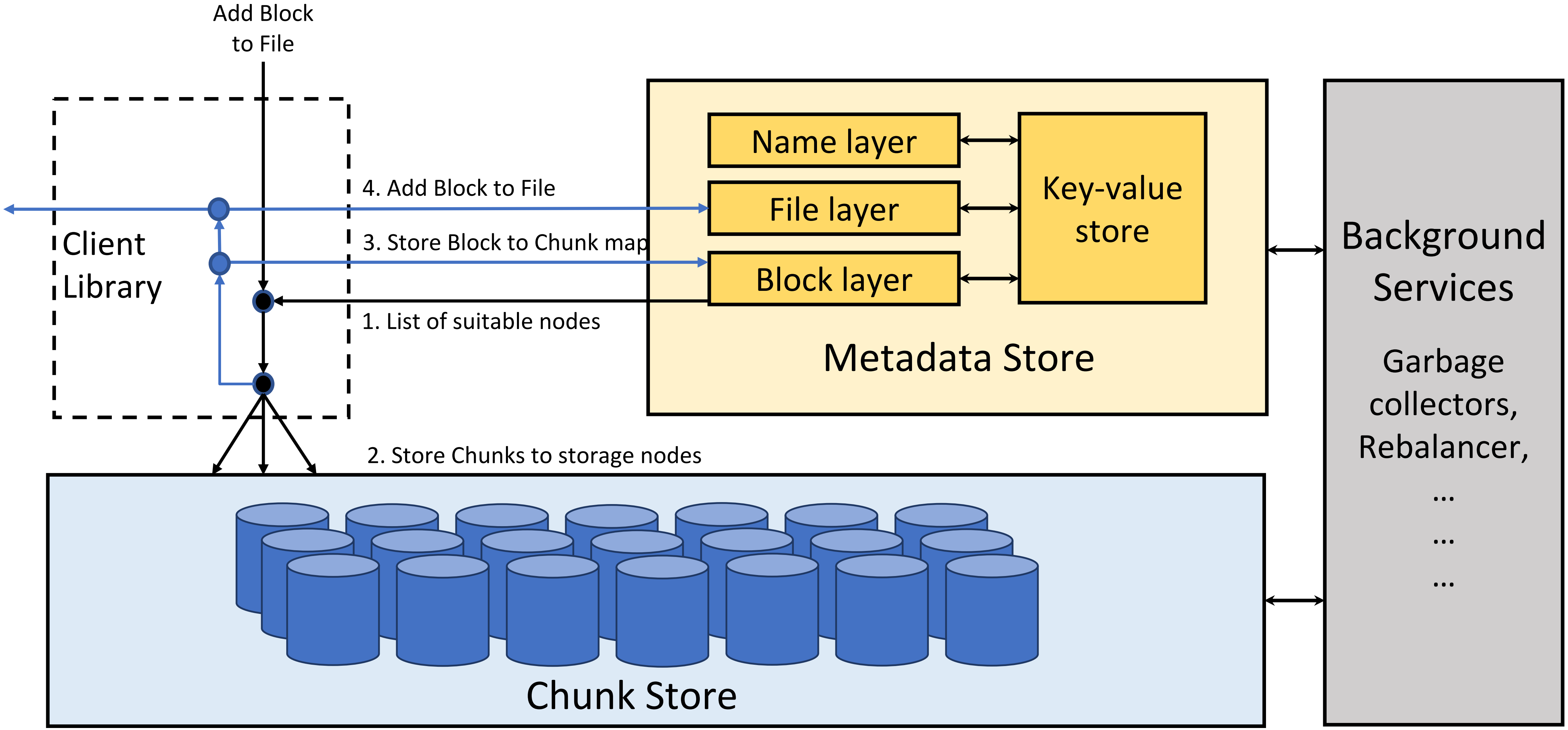
Scalability: Support Exabyte Scale Clusters



Scalability: Support Exabyte Scale Clusters



Scalability: Support Exabyte Scale Clusters



Performance: Match Specialized Systems

- Specialized storage systems optimize for the specific access pattern and performance requirements
- Tectonic uses *tenant-specific optimizations* to match the performance of specialized systems
- Optimizations are enabled by the Client Library, which runs in application binary
- Client library allows flexible and varying composition of Tectonic operations, which can be configured according to the needs of the tenant

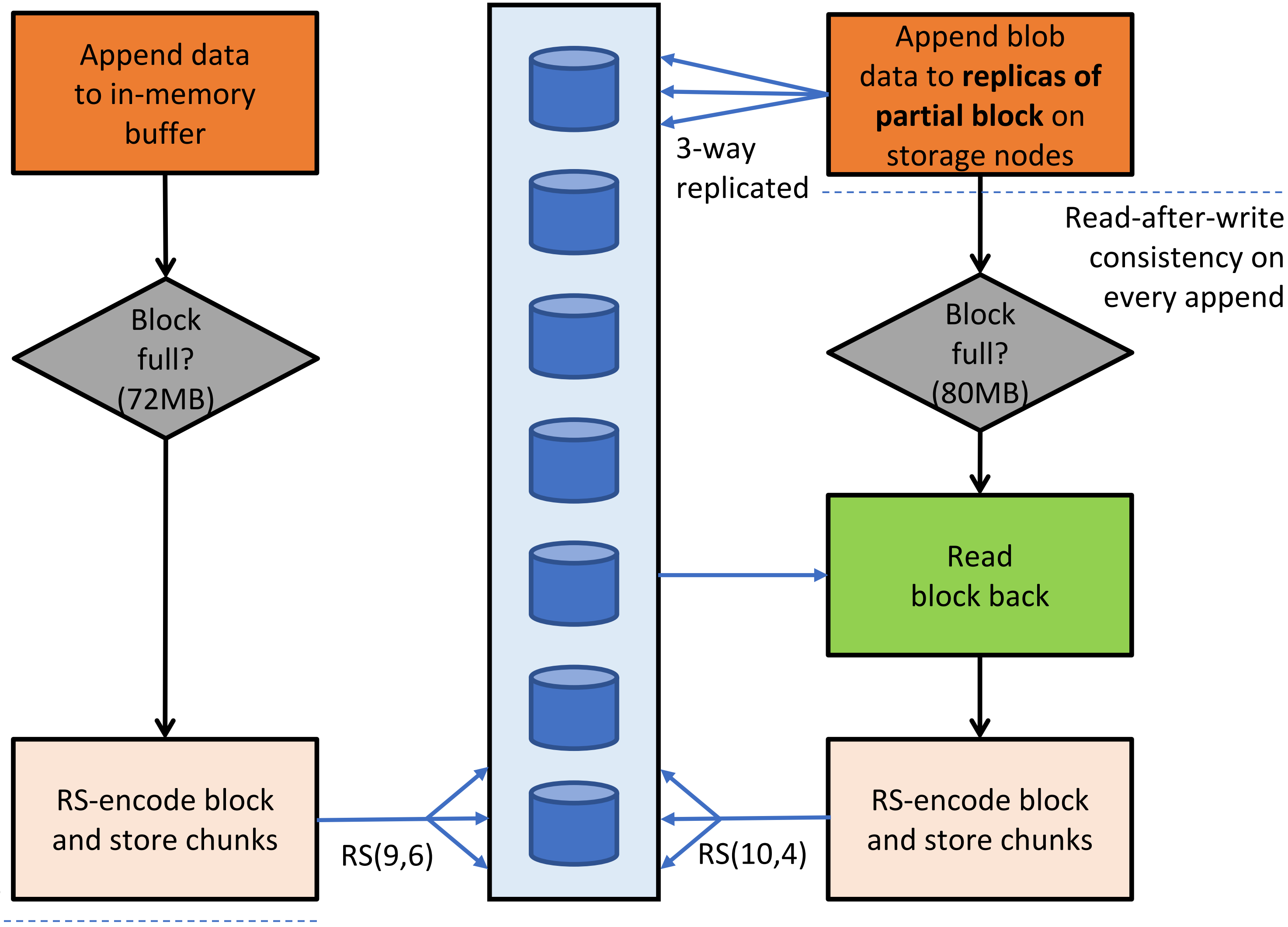
Tenant-specific Optimizations: Appends

Data warehouse

Files are large (100s of MBs)

Files are read after the creator closes the file

Minimize bytes written to store file to improve overall throughput



Append data to in-memory buffer
Block full? (72MB)
RS-encode block and store chunks
RS(9,6)
Read-after-write consistency only after file close
Append blob data to **replicas of partial block** on storage nodes
3-way replicated
Read-after-write consistency on every append
Block full? (80MB)
Read block back
RS-encode block and store chunks
RS(10,4)

Blob storage

Blob sizes are small (100s of KBs)

Blobs appended to log structured file

Blobs need to be persisted before acknowledging upload

Minimize latency for blob uploads, Later optimize storage

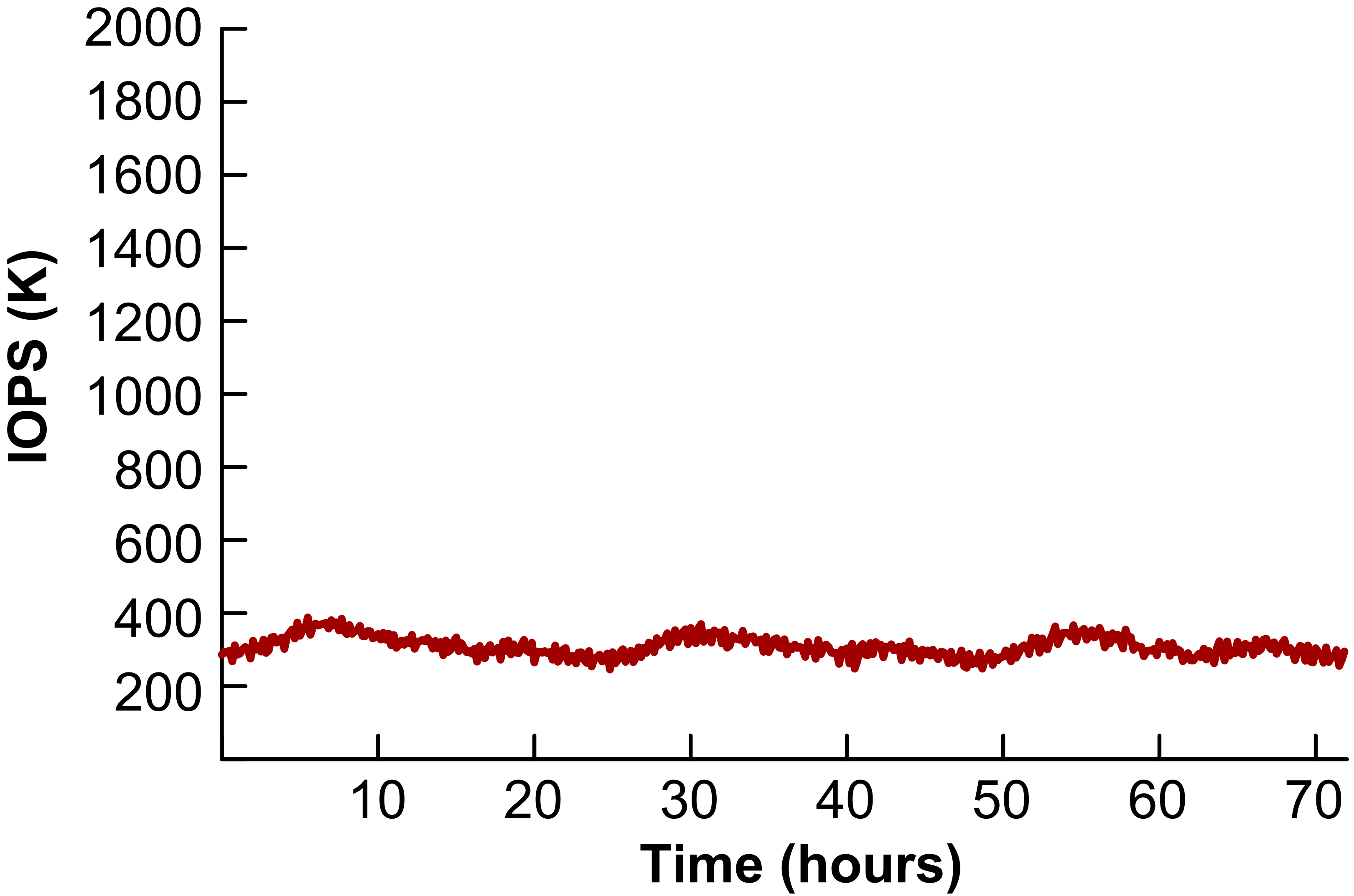
Results

- Tectonic clusters are ~10x the size of HDFS clusters, which simplifies production operations

- Blob storage latency in Tectonic comparable to Haystack

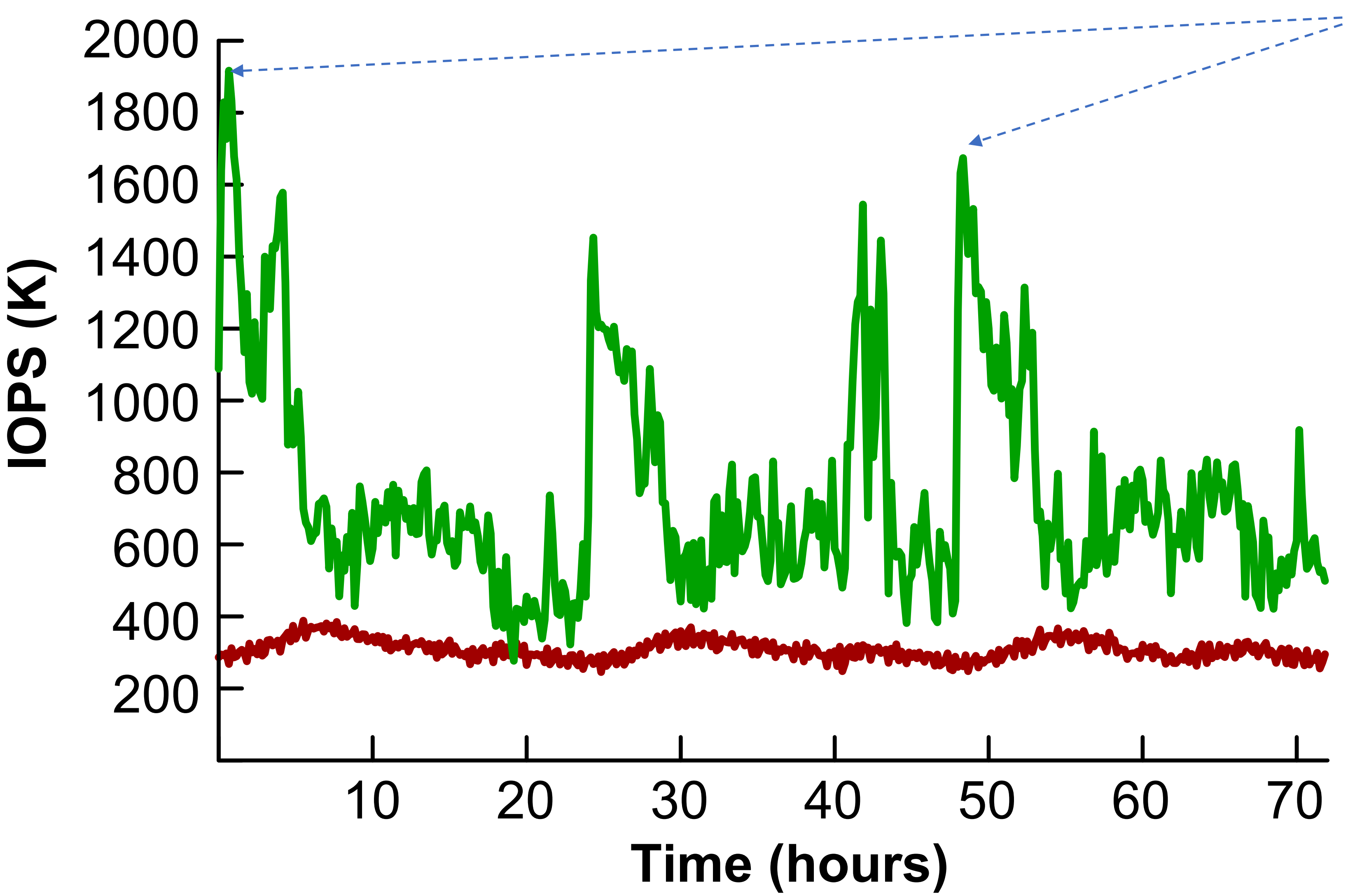
- In a multitenant cluster, data warehouse uses surplus IO from blob storage to serve its peaks

Efficiency From Storage Consolidation



Blob storage:
surplus IO available

Efficiency From Storage Consolidation



Data warehouse:
peaks need excess IO,
use surplus from
blob storage

Blob storage:
surplus IO available

Tectonic Provides Datacenter-Scale Storage

- Replaced previous constellation of specialized storage systems
  - Simpler operations
  - Better resource utilization

- Tectonic's design addresses the key challenges:
  - Scalability: disaggregated linearly scalable components
  - Performance: tenant-specific optimizations via client library
  - …

Thank You