BatchFS++

In search of the right way for extreme-scale HPC file system metadata

Qing Zheng[1], Kai Ren[1], Garth Gibson[1], Bradley W. Settlemyer[2]

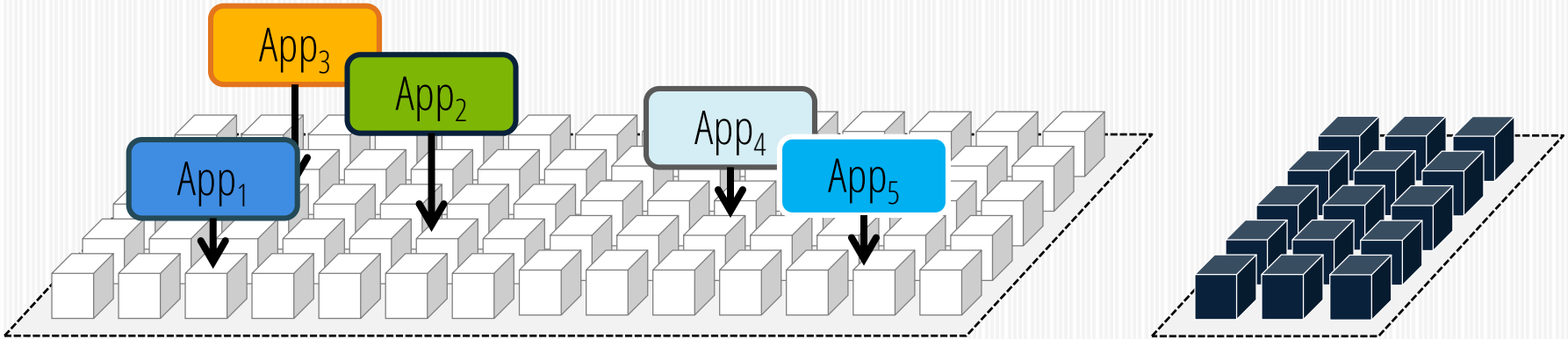
[1]Carnegie Mellon University

[2]Los Alamos National Laboratory

[LA-UR-15-25703]

HPC defined by …

- parallel scientific applications
  - running on compute nodes with pooled storage for highly parallel I/O

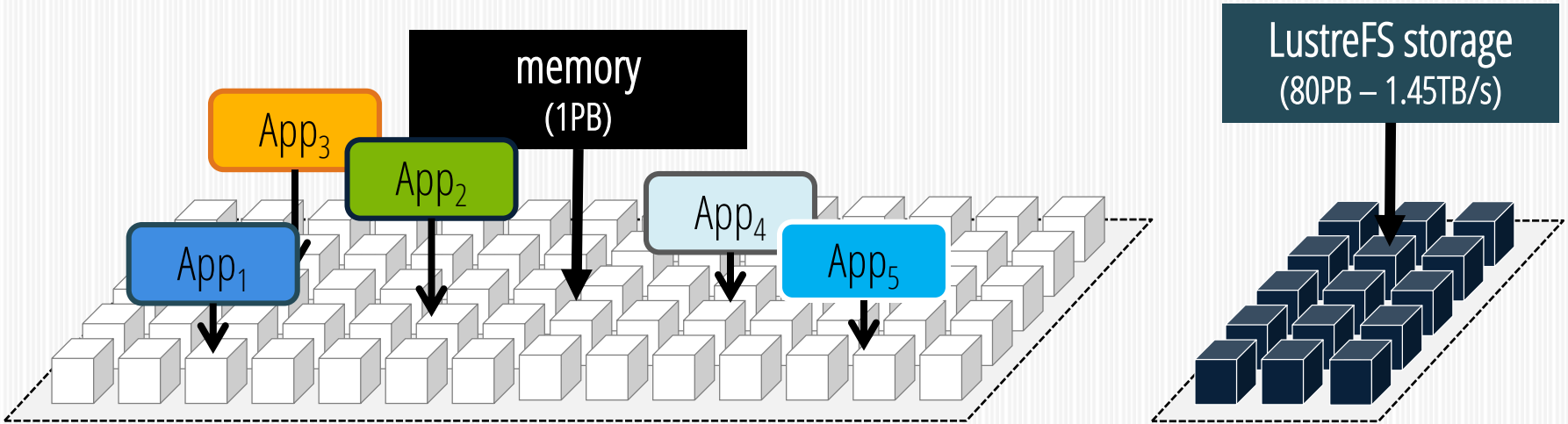


compute nodes w/ fast interconnect
(10,000+)

pooled storage nodes
(100+)

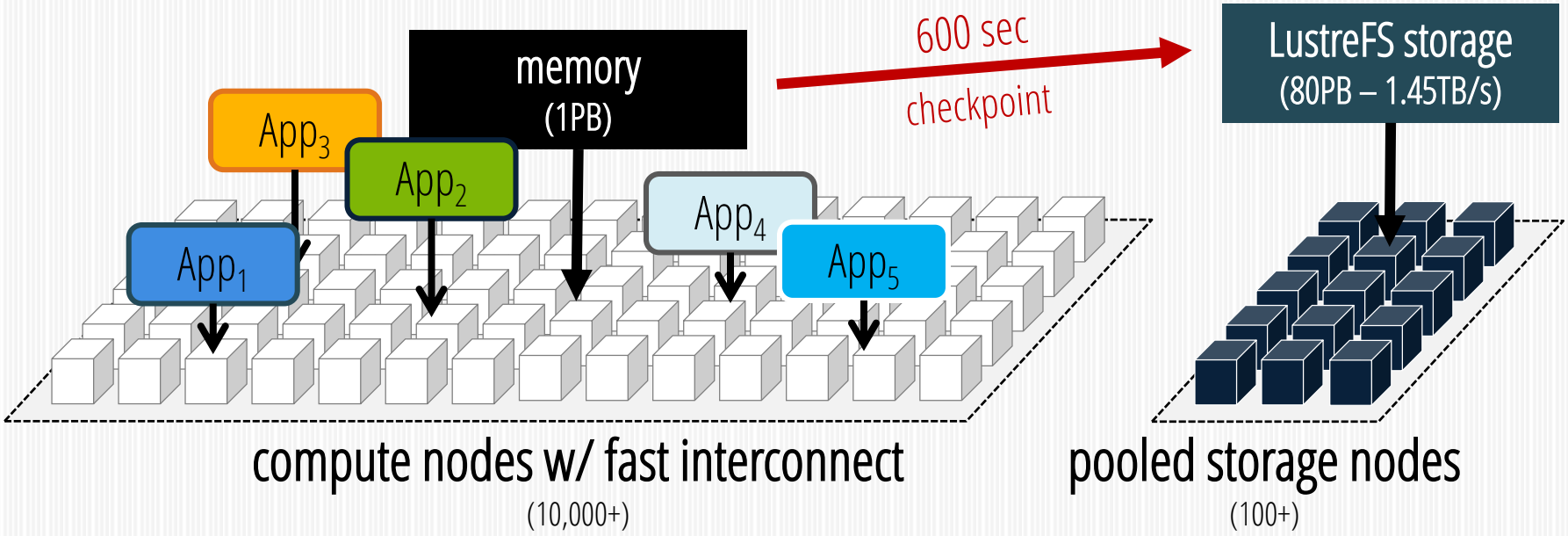
HPC Checkpointing

- copying memory from compute nodes to a backend file system



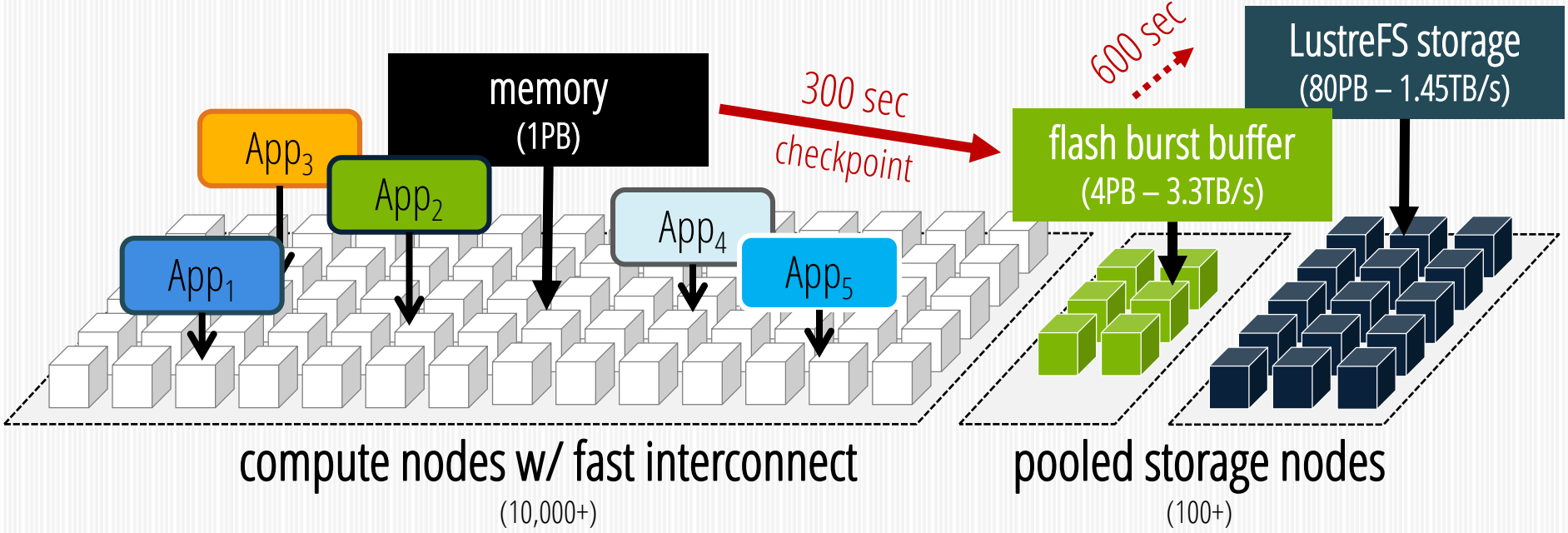
HPC Checkpointing

- copying memory from compute nodes to a parallel file system



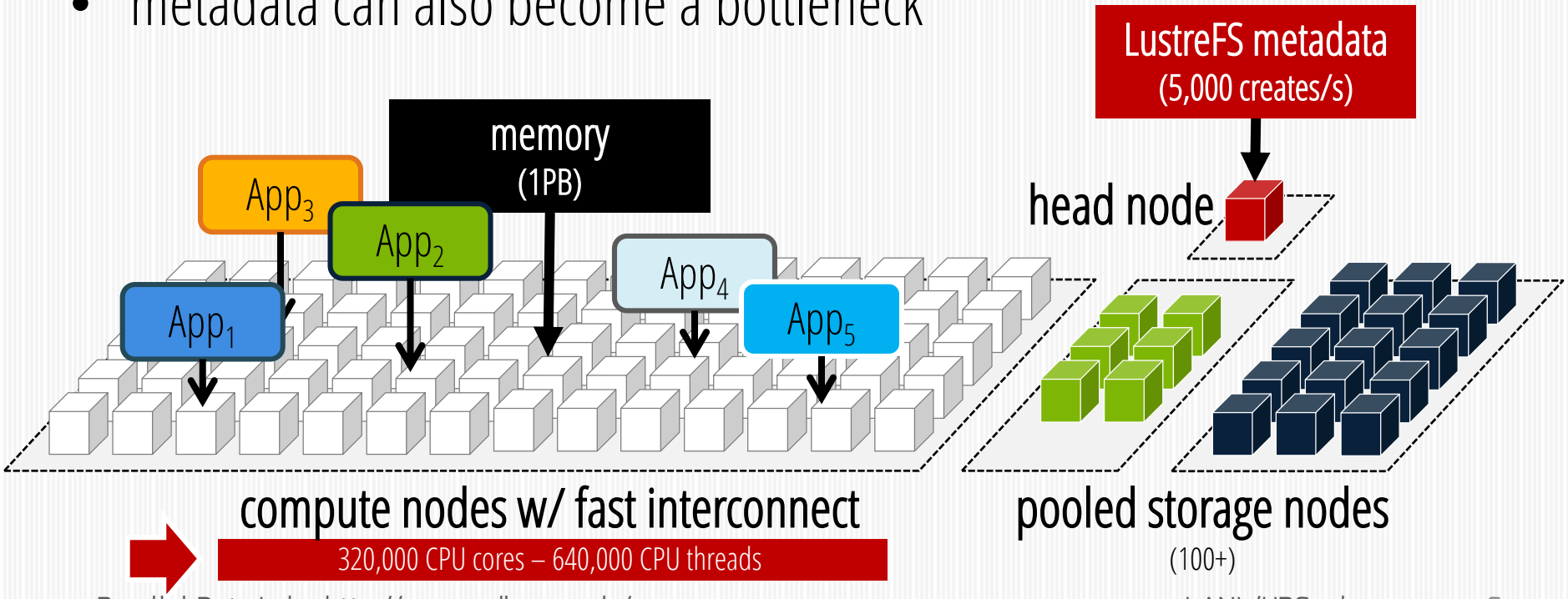
HPC Checkpointing

- two-tier storage for higher bandwidth



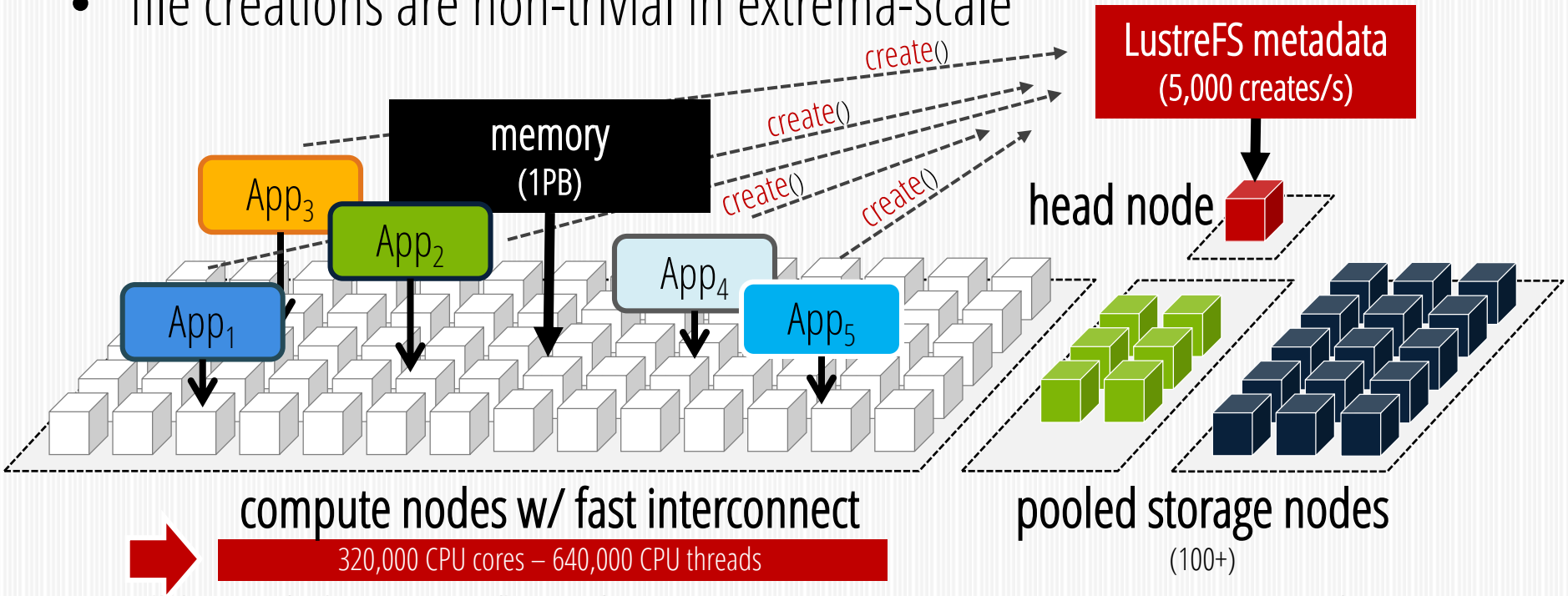
HPC Checkpointing

- metadata can also become a bottleneck



HPC Checkpointing

- file creations are non-trivial in extrema-scale

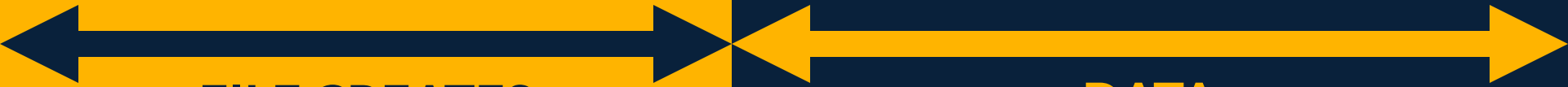



128s
300s
FILE CREATES
DATA

2X CPU cores and 2X storage bandwidth

256s

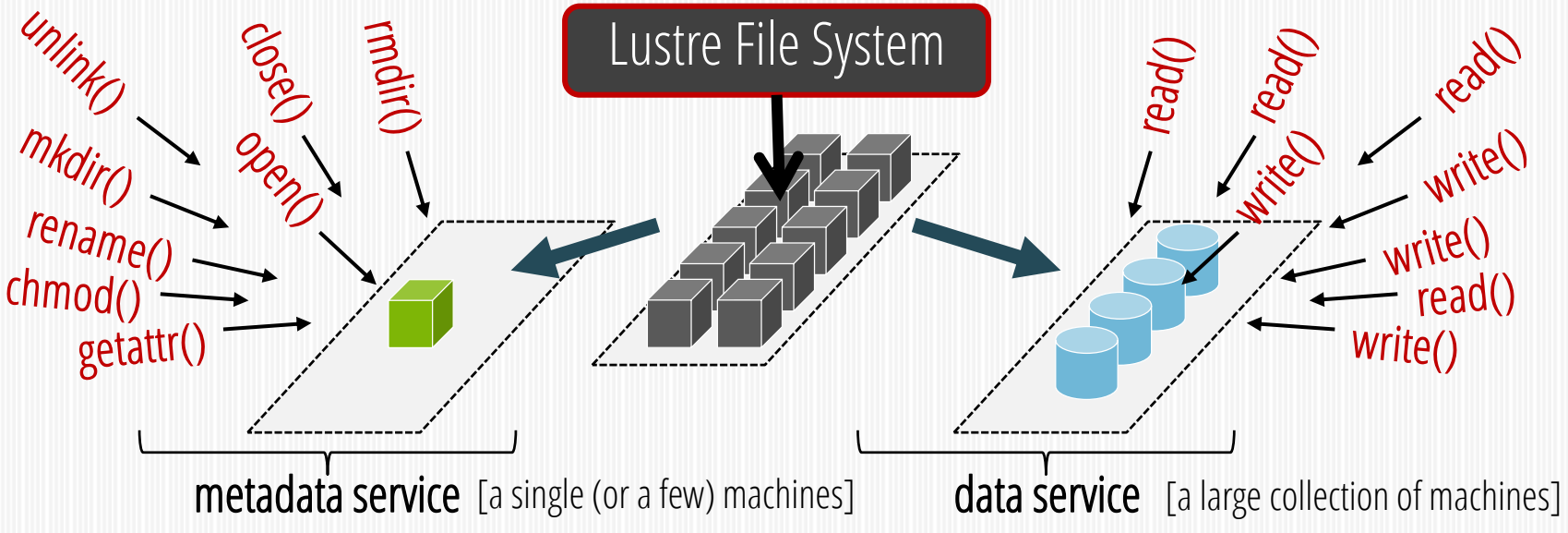
300s



FILE CREATES

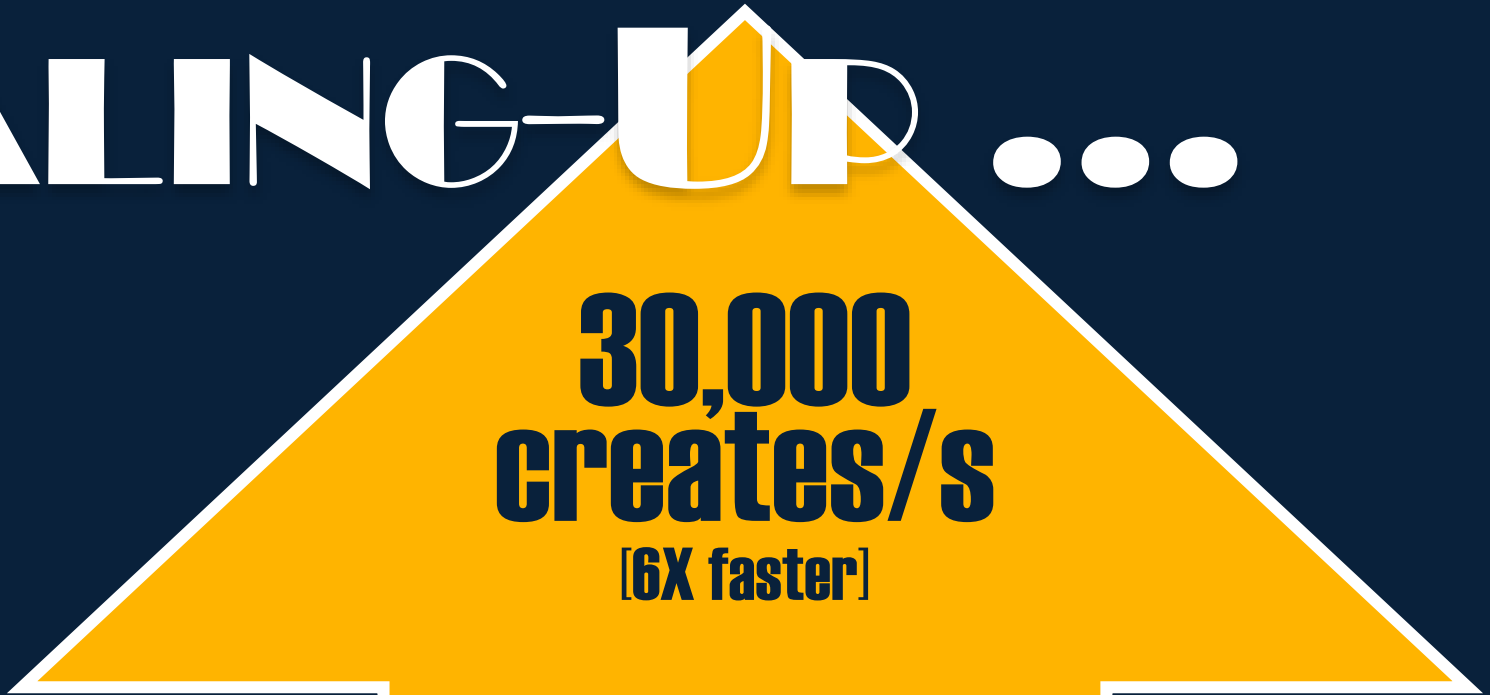
DATA

Bias for data …



metadata eventually a problem !!

Scaling-Up …



Harden Lustre File System

Scaling-Up …

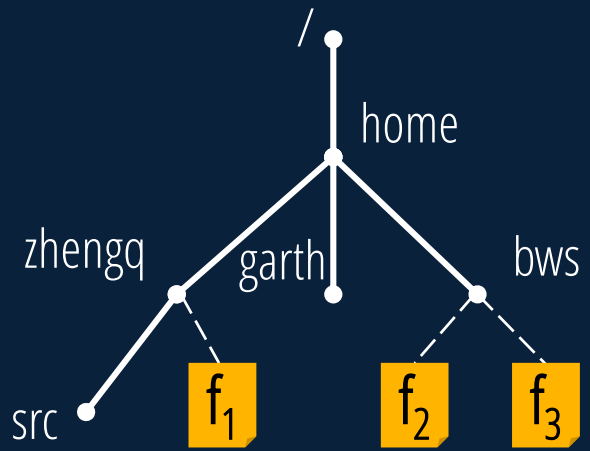
expensive w/ limited improvements

Scaling-Out ...

Use multiple dedicated machines to serve file system metadata

DISTRIBUTED METADATA

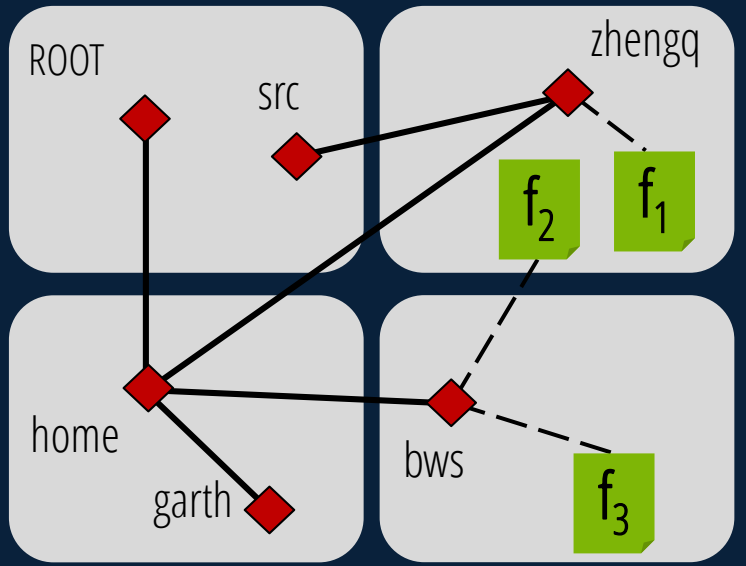
Scaling-Out ...



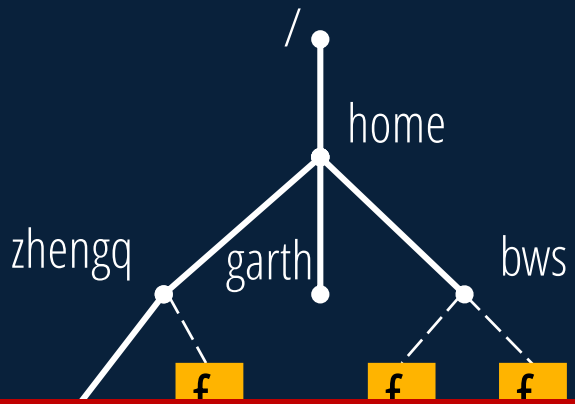
hierarchical directory structure

IndexFS

dynamic directory
allocation & splitting

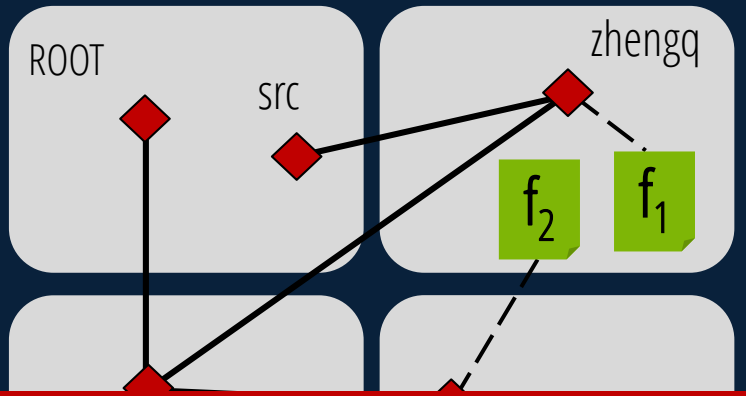


Scaling-Out ...



IndexFS

dynamic directory allocation & splitting



requires a large # of dedicated machines

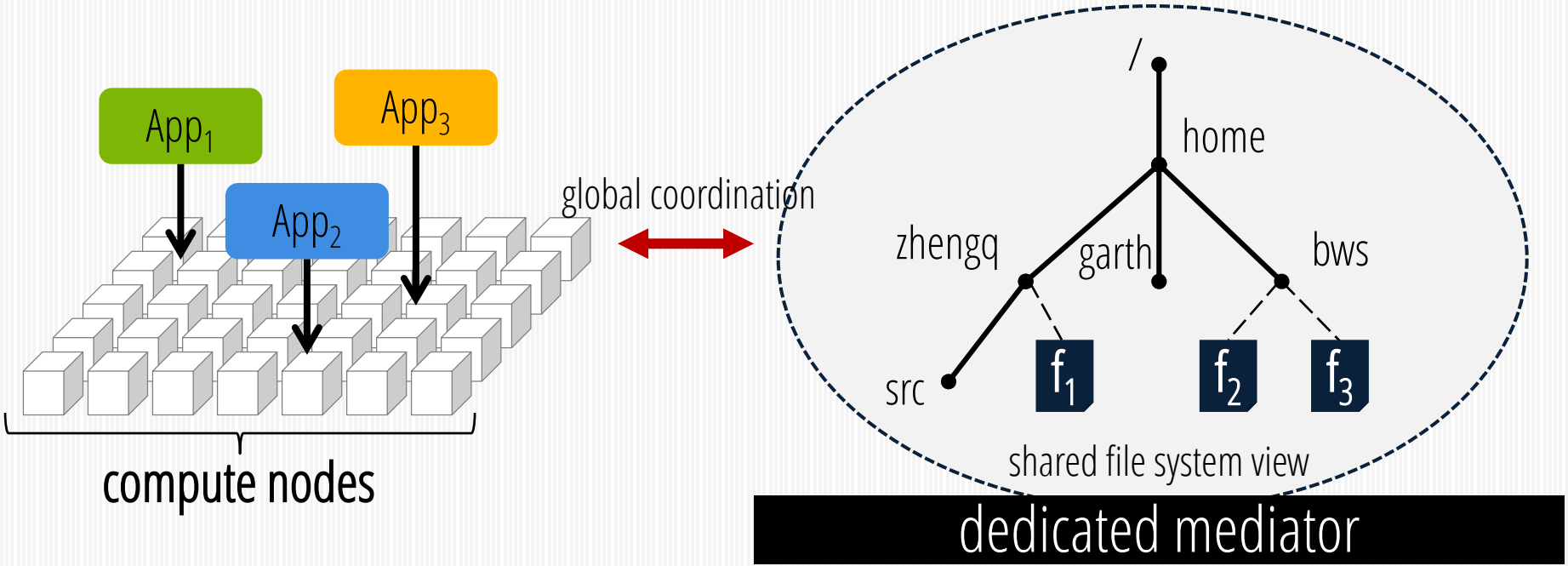
GOAL

Cost-effective highly-parallel metadata

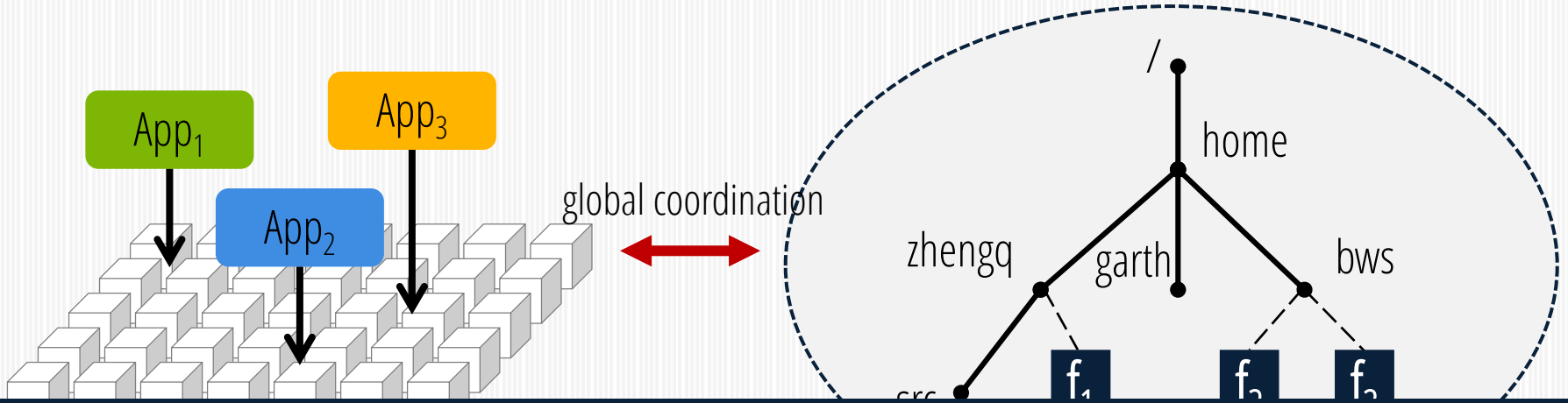
Orders of magnitude faster then LustreFS


BatchFS++

Traditional Metadata Model



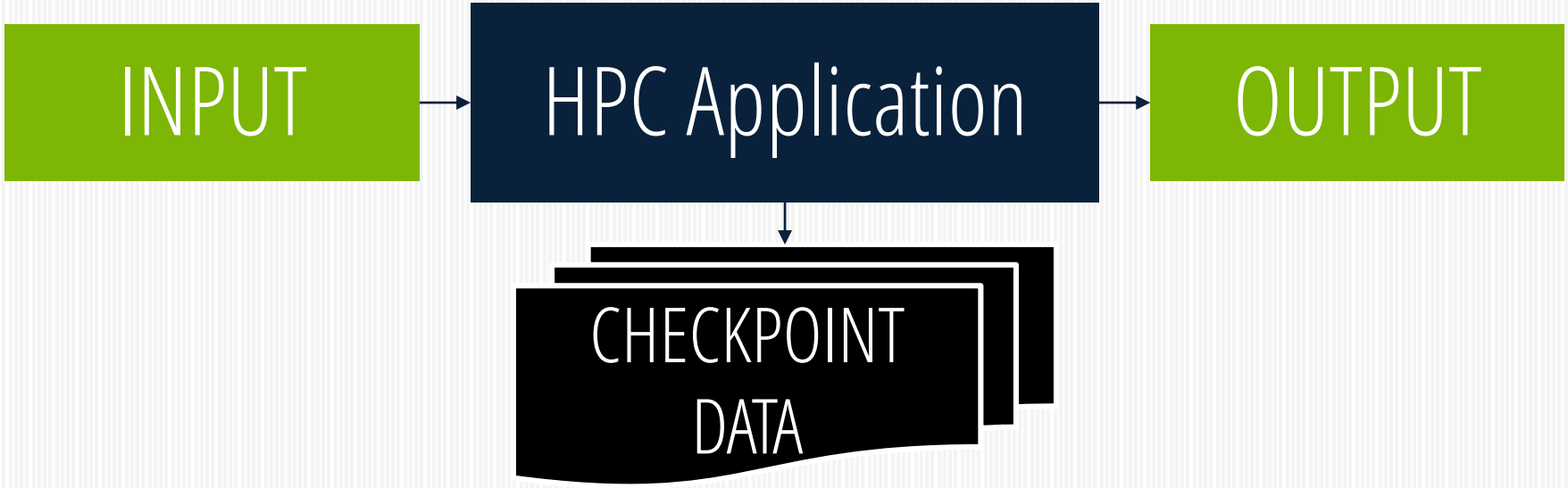
Traditional Metadata Model



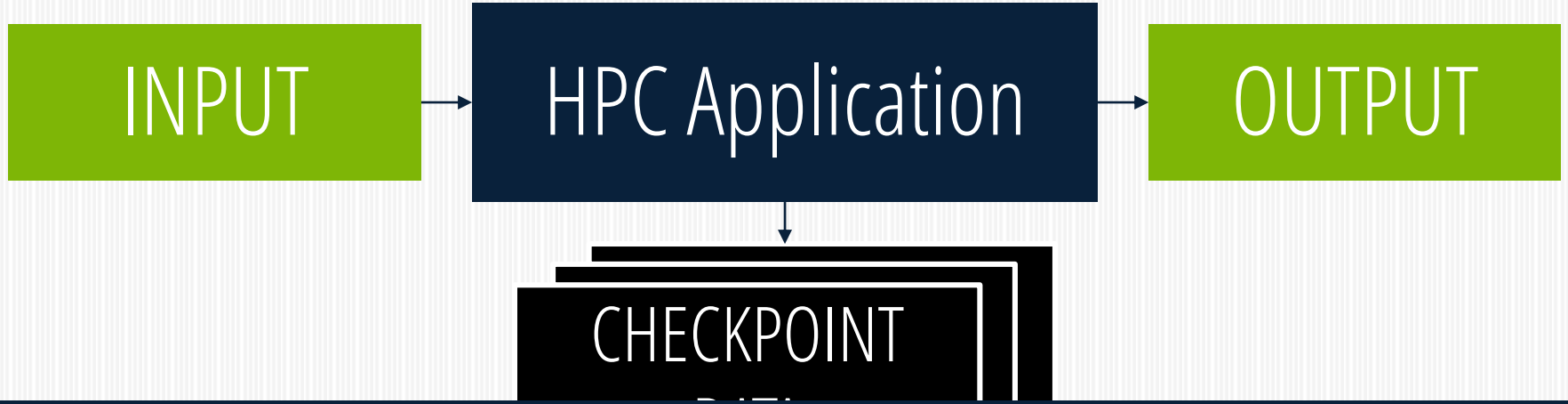
A single authoritative view of the file system

Allows different apps to shared data and achieve synchronization

HPC I/O is much simpler …



HPC I/O is much simpler …



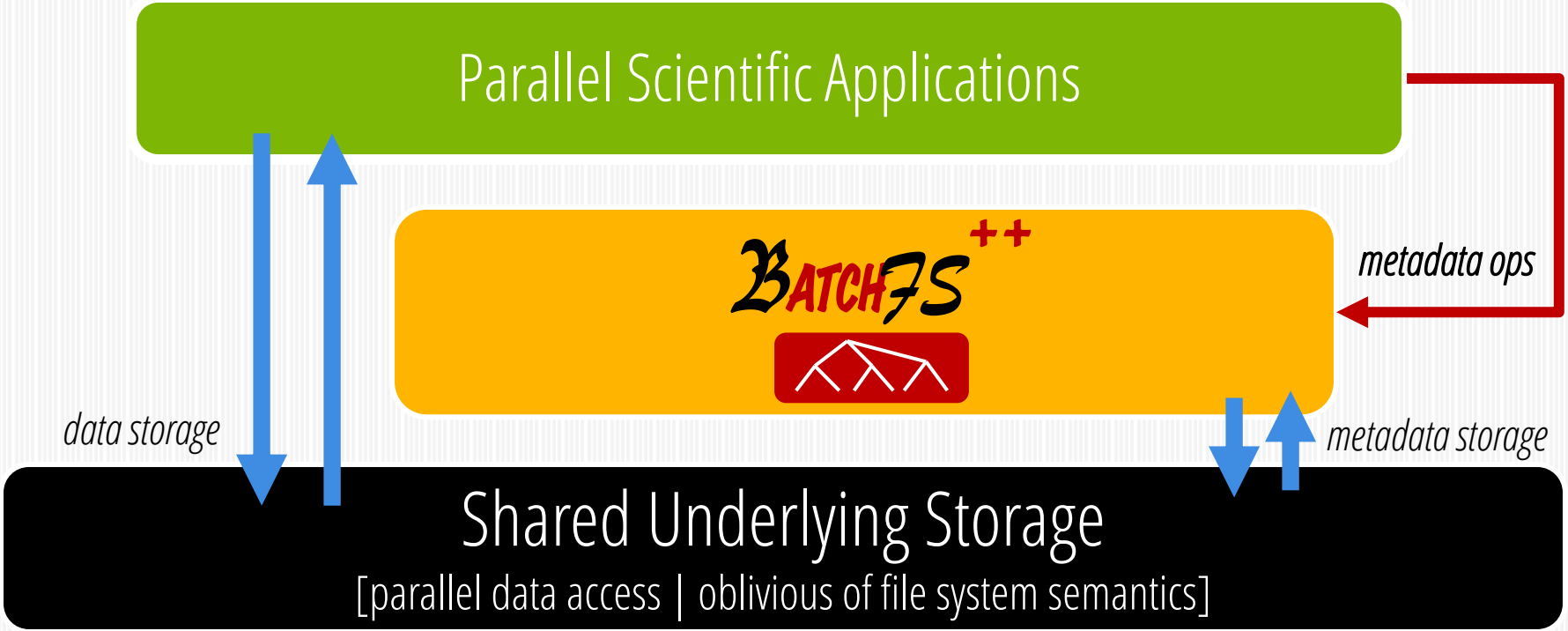
HPC applications don't need the FS to achieve synchronization
Different apps operate on different sets of files

HPC I/O is much simpler …

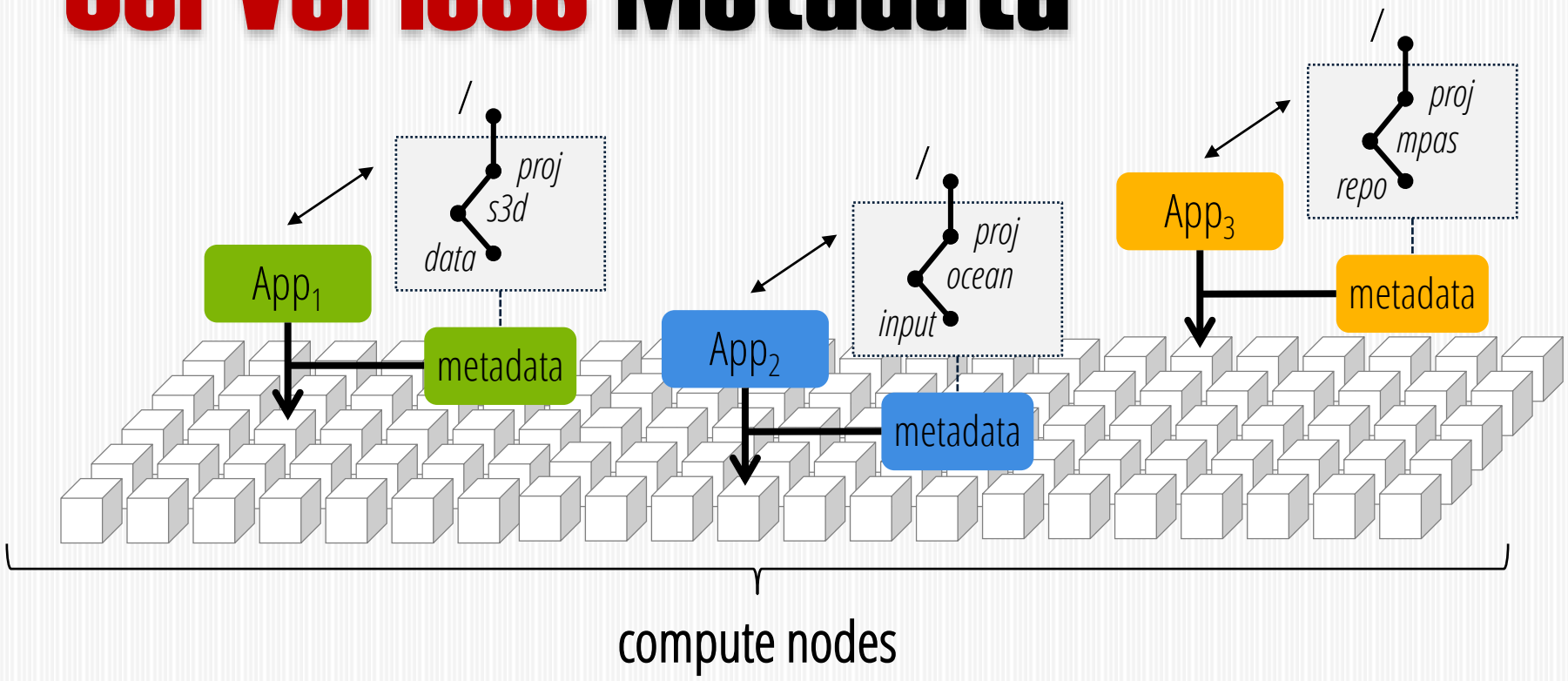
A single unified file system view overkill for HPC applications
Global coordination via a dedicated service bad for performance

HPC applications don't need the FS to achieve synchronization
Different apps operate on different sets of files

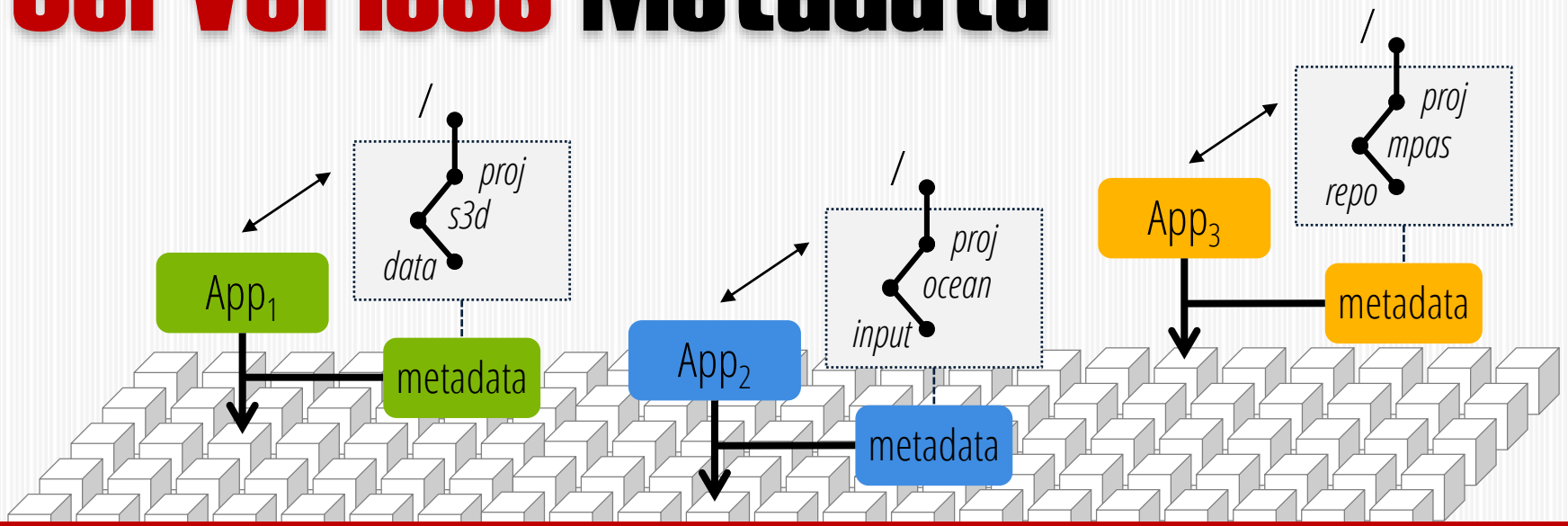
Middleware Design



Serverless Metadata



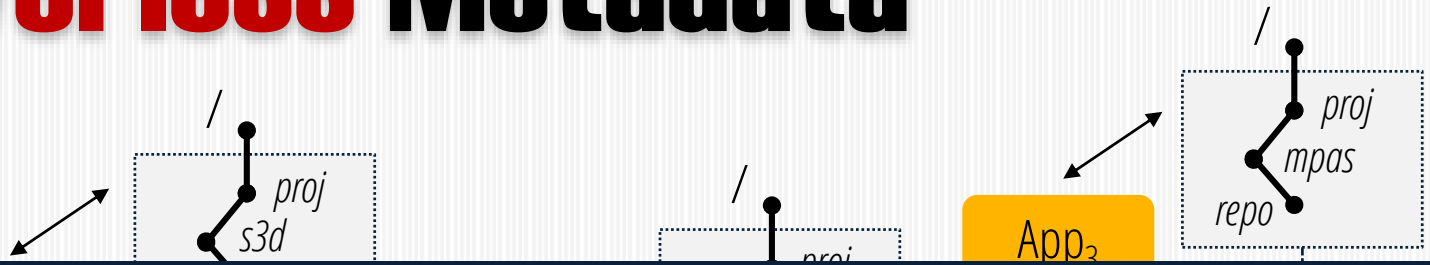
Serverless Metadata



No such global namespace managed by a centralized mediator

Each app only communicates with its private metadata servers

Serverless Metadata



Fundamental Assumption

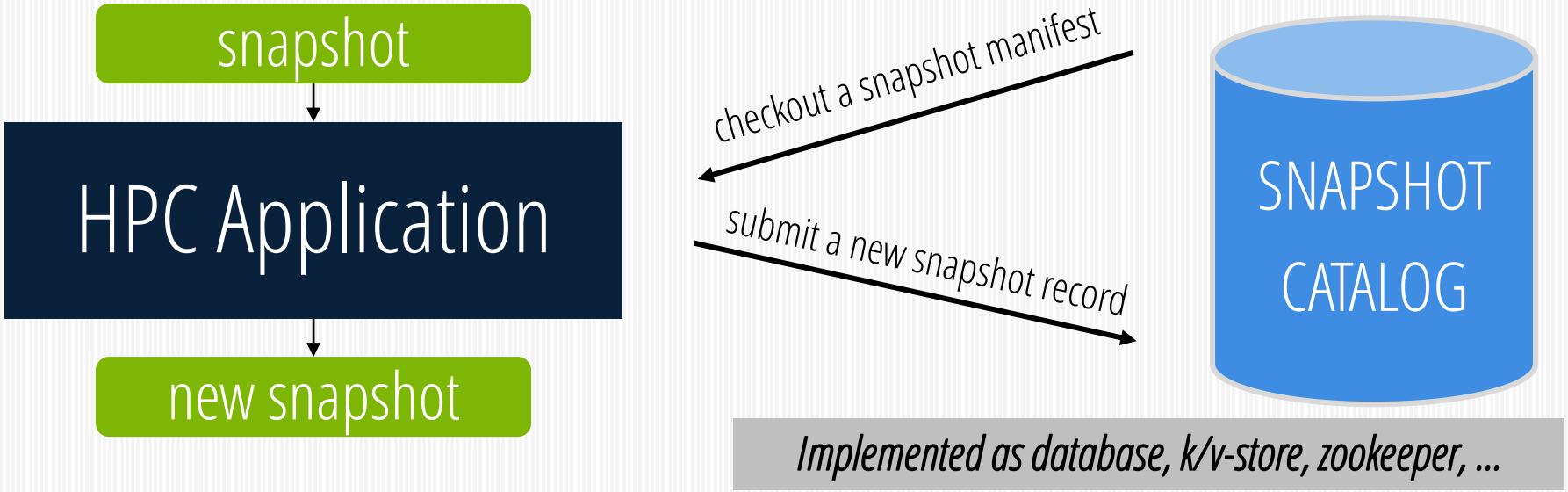
HPC applications don't communicate with each other

No such global namespace managed by a centralized mediator

Each app only communicates with its private metadata servers

App Execution Model

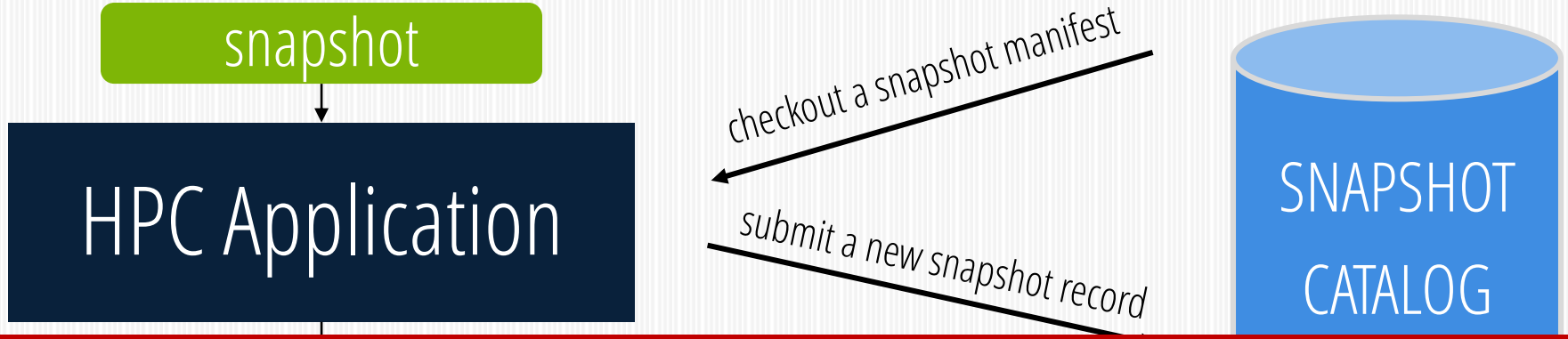
[snapshot = static view of a file system]



Implemented as database, k/v-store, zookeeper, ...

App Execution Model

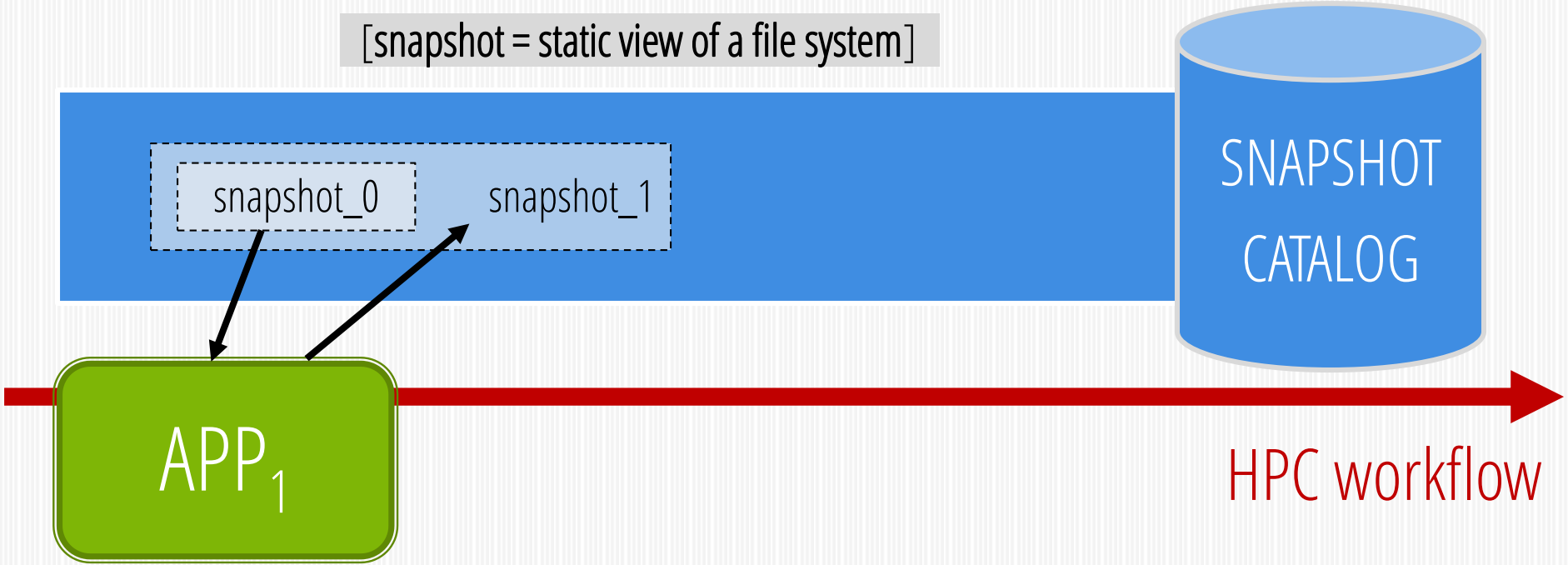
[snapshot = static view of a file system]



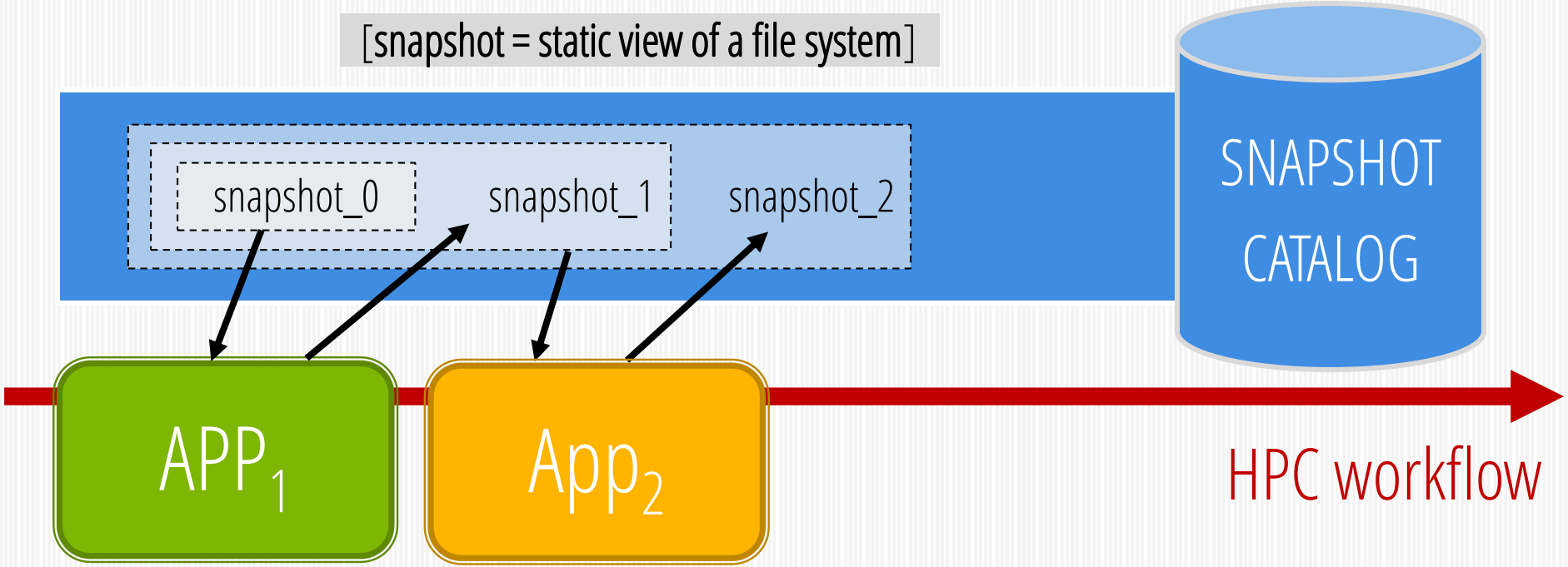
Keeps each app isolated from other running apps

Truly parallel metadata processing among different apps

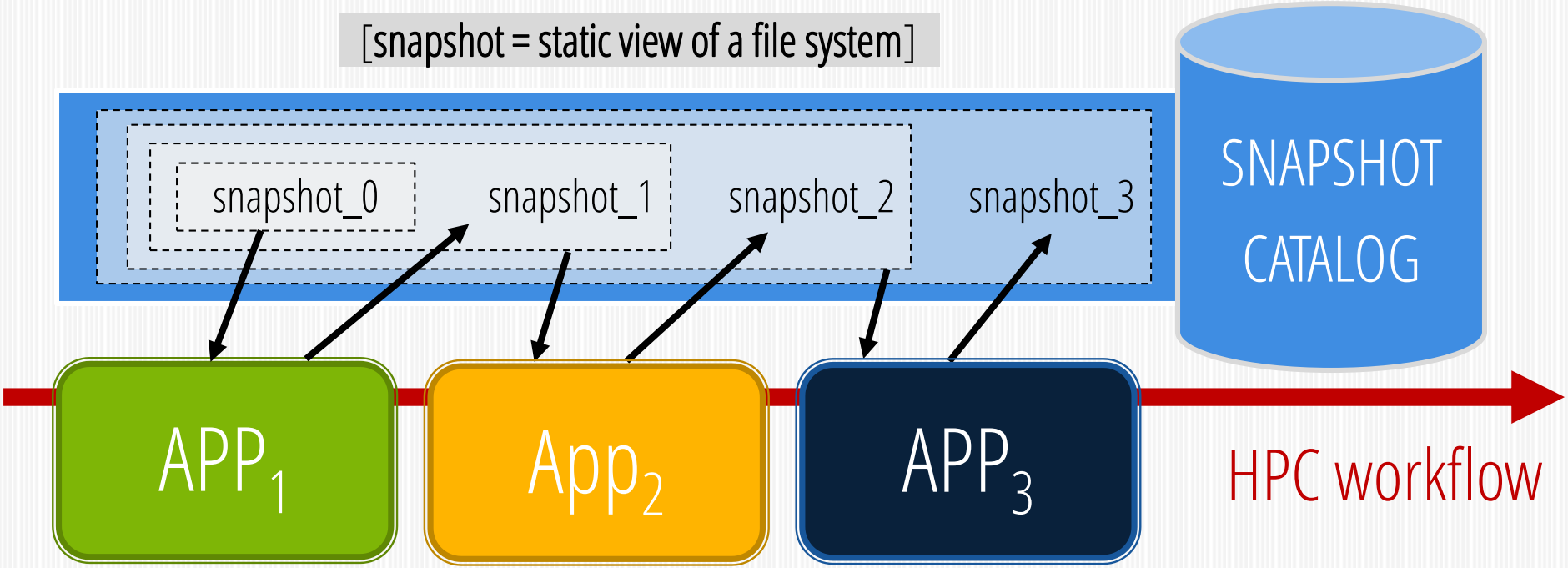
Sharing Data Between Apps



Sharing Data Between Apps

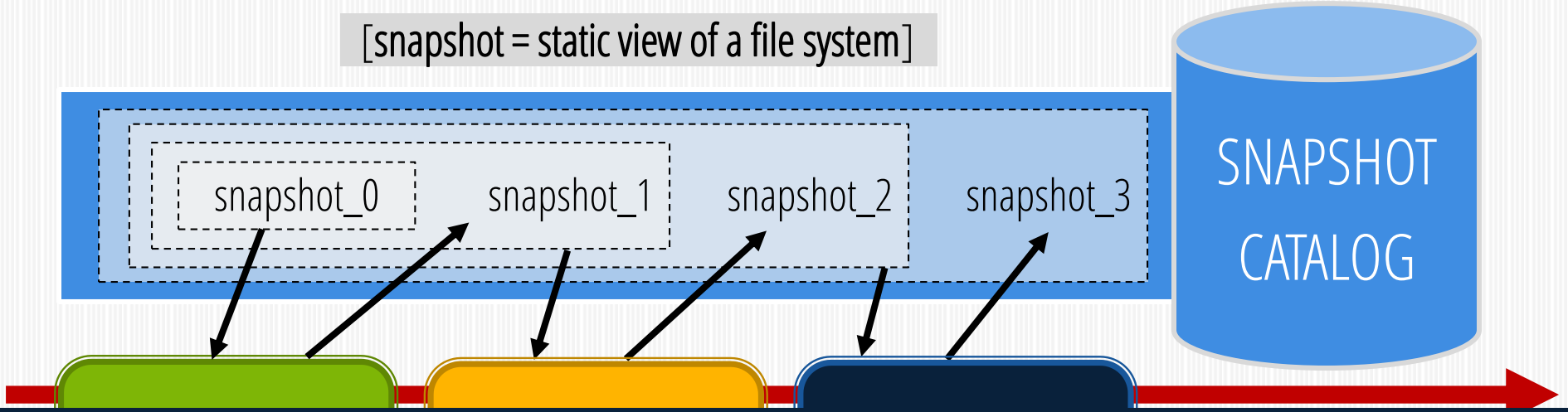


Sharing Data Between Apps



Sharing Data Between Apps

[snapshot = static view of a file system]

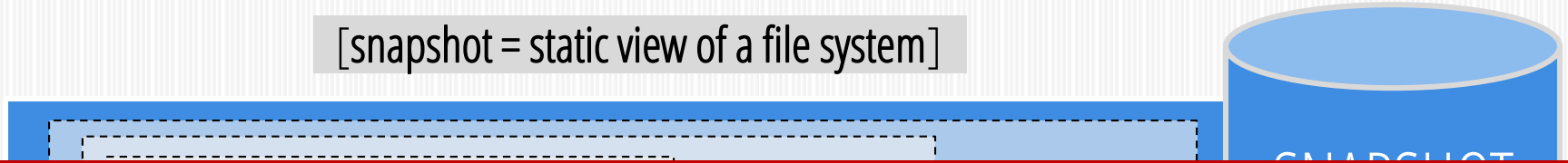


Converts online to batched offline sharing

Allows apps to decide file visibility and timings for communication

Sharing Data Between Apps

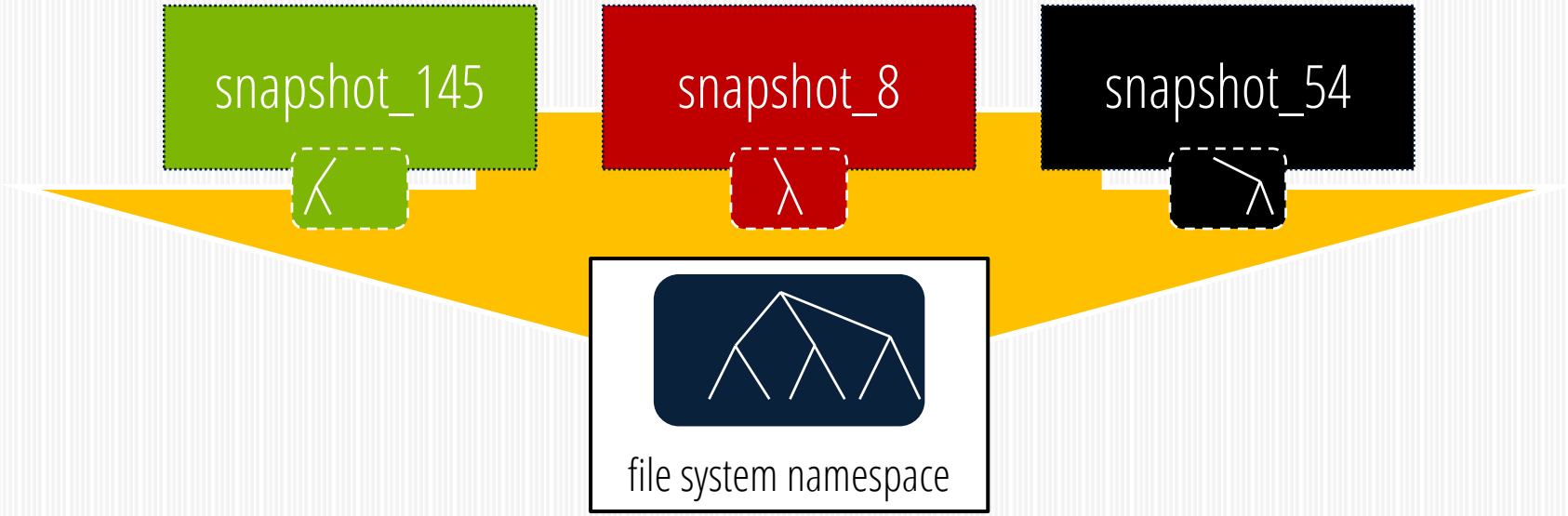
[snapshot = static view of a file system]



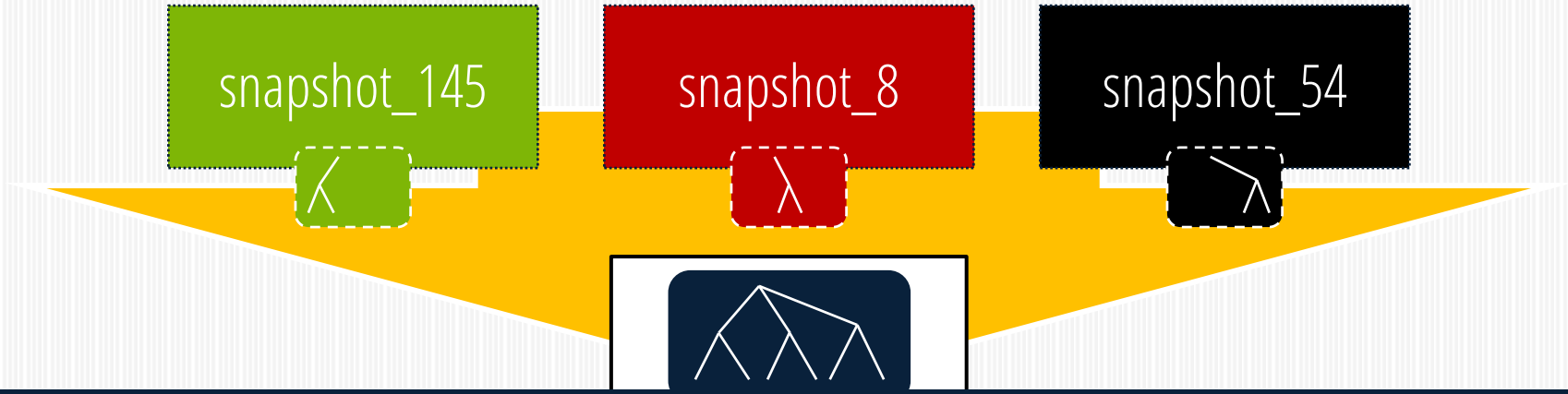
Avoids unnecessary coordination
synchronously enforced by a centralized metadata service

Converts online to batched offline sharing
Allows apps to decide file visibility and timings for communication

Snapshot Merging



Snapshot Merging

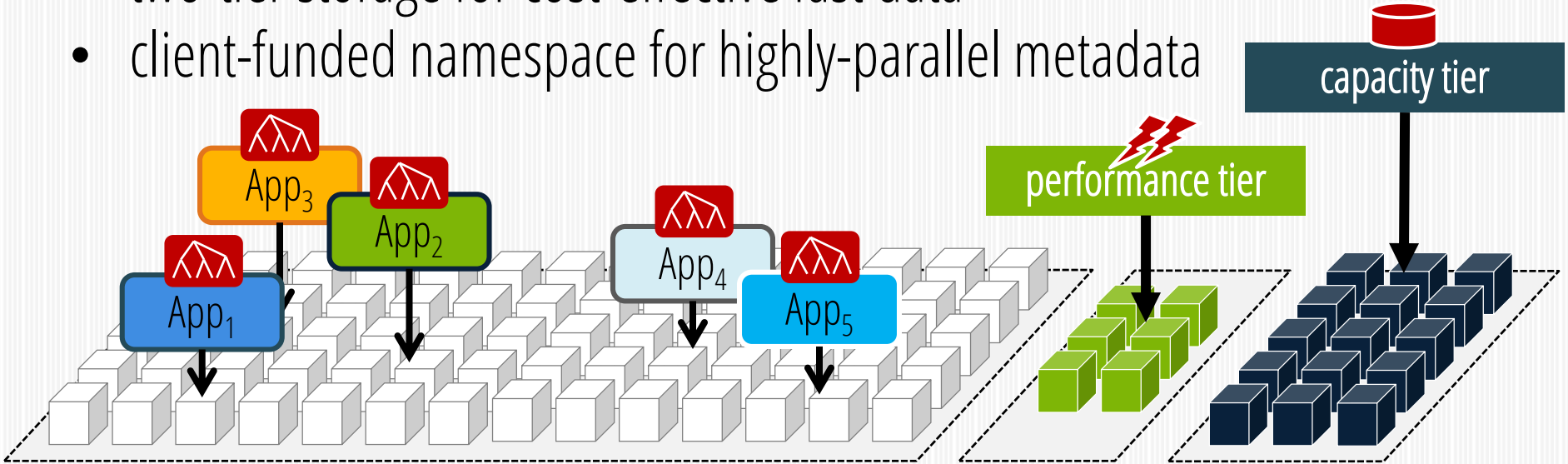


Conflicts resolved at read path to avoid unnecessary computation

Resolution performed by apps instead of a super authority

HPC in 10 years …

- two-tier storage for cost-effective fast data
- client-funded namespace for highly-parallel metadata



Reference

BatchFS | Scaling the File System Control Plane with Client-Funded Metadata Servers (PDSW14)

IndexFS | Scaling File System Metadata Performance with Stateless Caching and Bulk Insertion (SC14)

QUESTIONS