CloudDrive: Smarter Block Level Cloud-Backed Storage

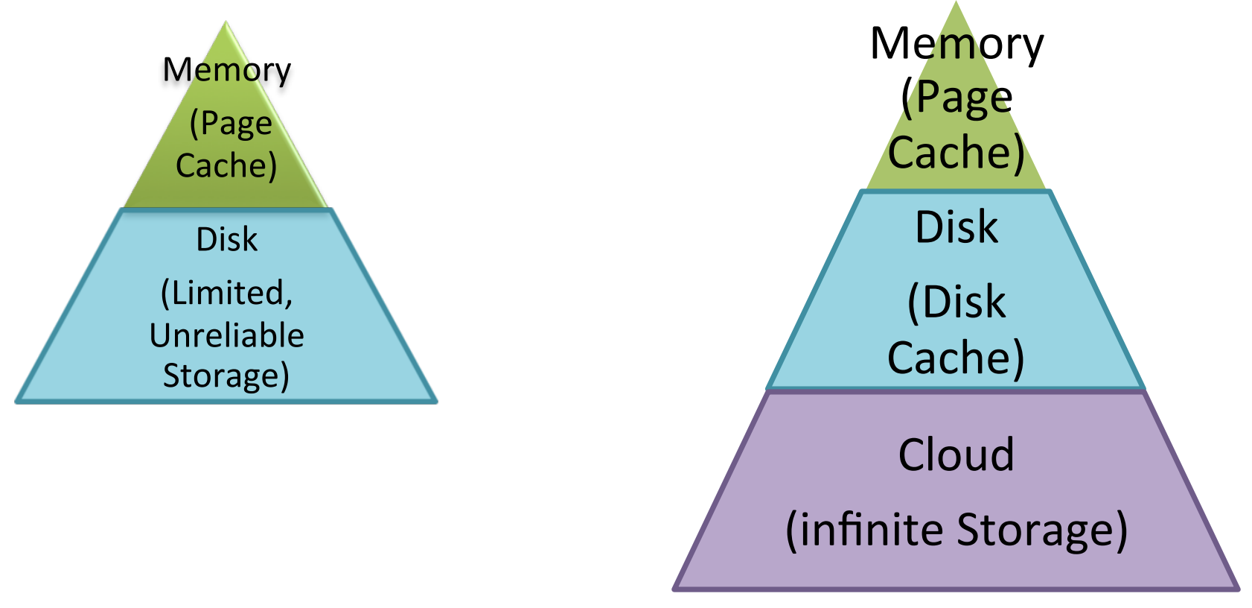
Ishani Ahuja, Suli Yang, Remzi H. Arpaci-Dusseau, Andrea C. Arpaci-Dusseau

Motivation

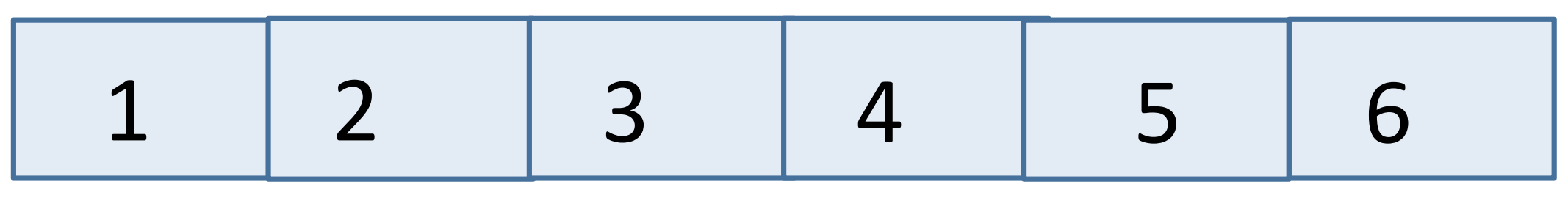
Cloud Storage at Block Level:
- Level below the disk in the storage hierarchy
- **Higher latency and higher storage capacity**
- Provides **reliability and mobility**
- Potential to further simplify file systems[1]

Why is Block Level interesting and hard?
- Backwards compatible with existing file system
- Lacks file system information and semantic inference at Block level is hard

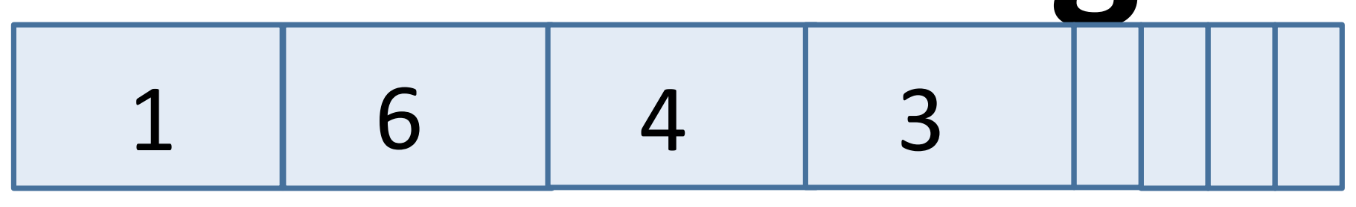
Why is Strawman Block Level Approach bad?
- Existing file systems are Cloud unaware and makes assumptions about underlying storage as disk. These assumptions affect the performance and capacity of the system heavily in case of a disk cache , cloud backed storage.



Basics of Block Level Cloud Storage

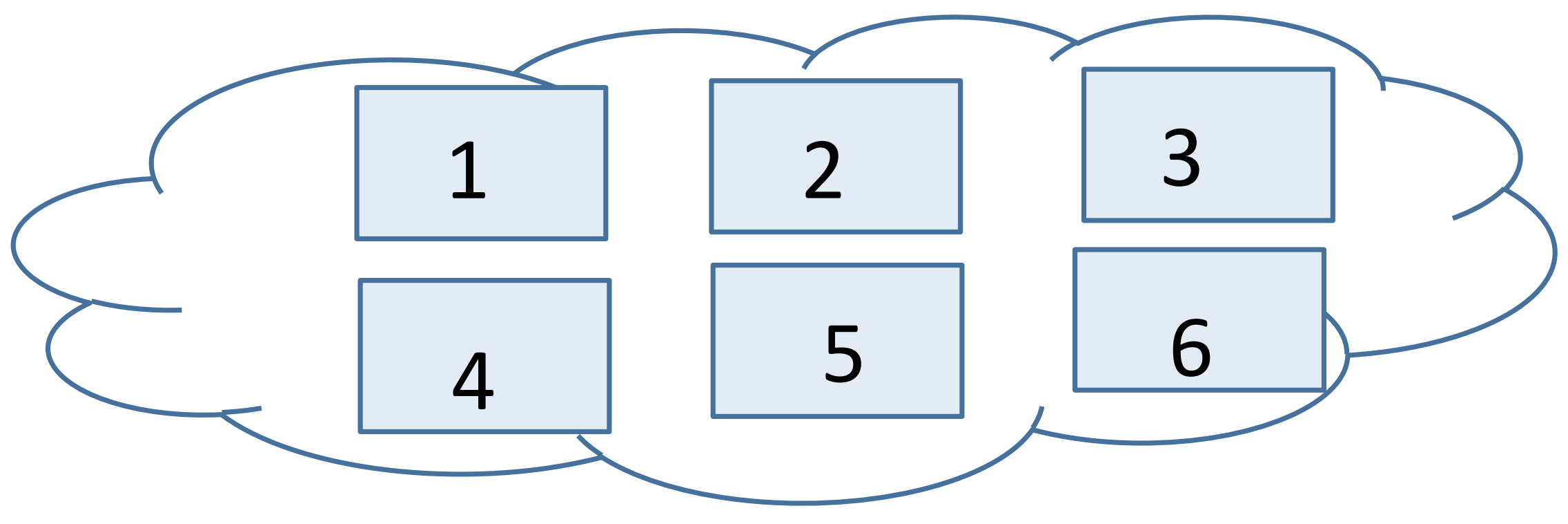


File System View of Storage

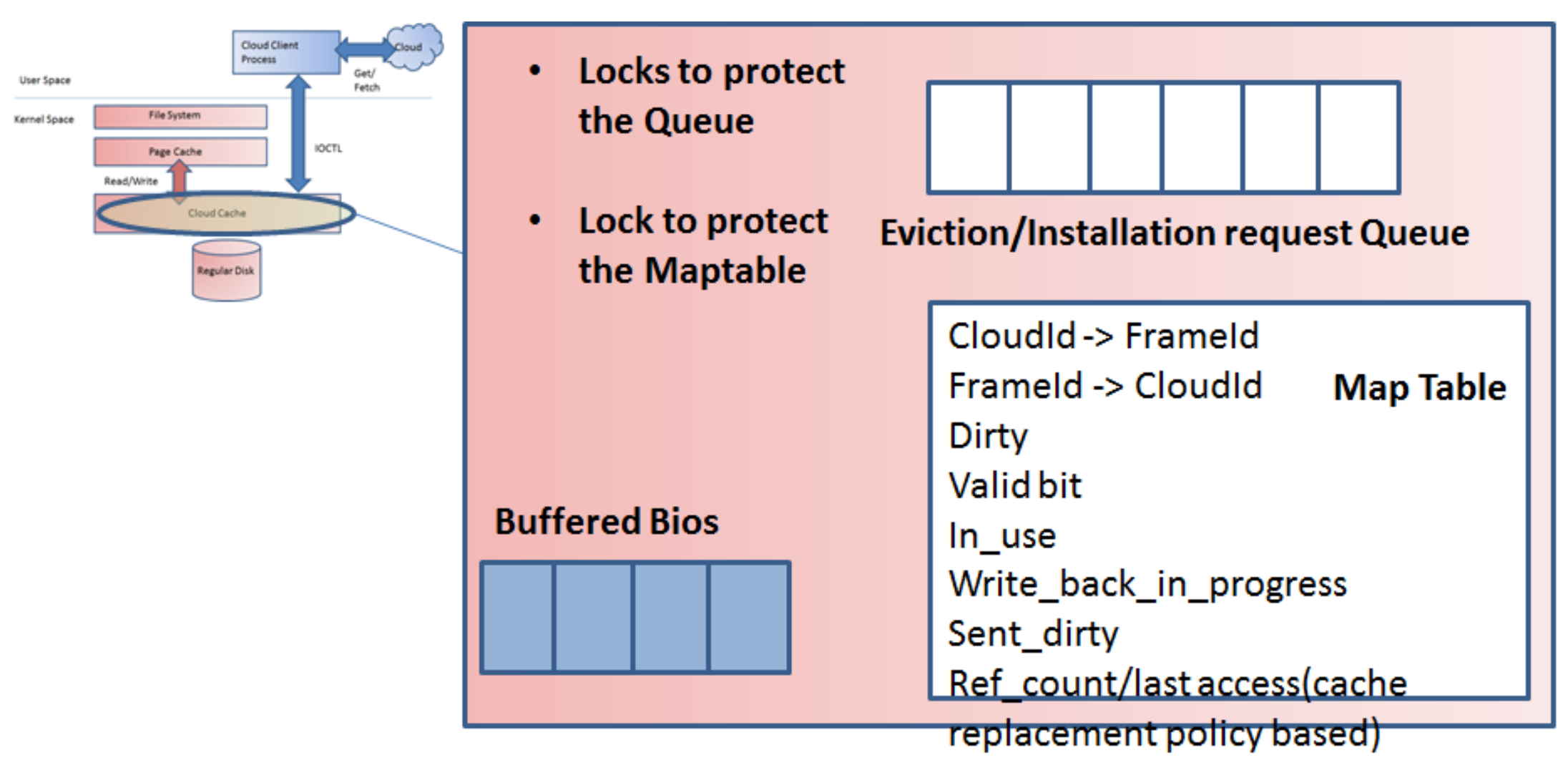


Disk as a cache

On – disk mapping table



Storage in Cloud Bucket in 4MB Files



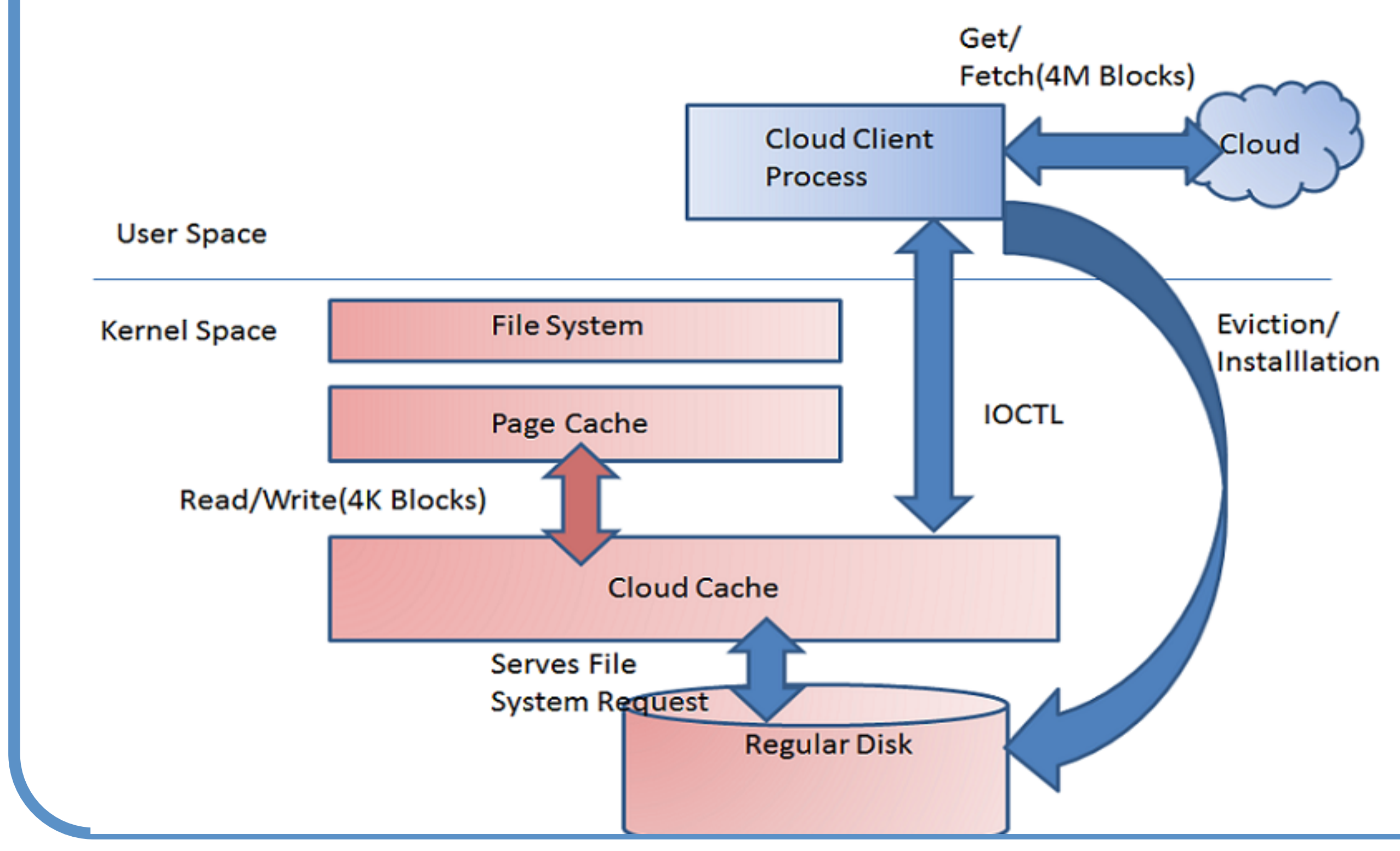
- Locks to protect the Queue
- Lock to protect the Maptable

CloudId -> FrameId
FrameId -> CloudId
Dirty
Valid bit
In_use
Write_back_in_progress
Sent_dirty
Ref_count/last access(cache replacement policy based)

in-memory state

Disk is used as a cache . **Cloud Blocks** are stored in **Disk Frames.** The Mapping information (BlockId -> FrameId) is stored in a map-table with other **in-memory state. The Mapping Information** is persisted in the disk cache to allow usage of cached data **across system reboot.**

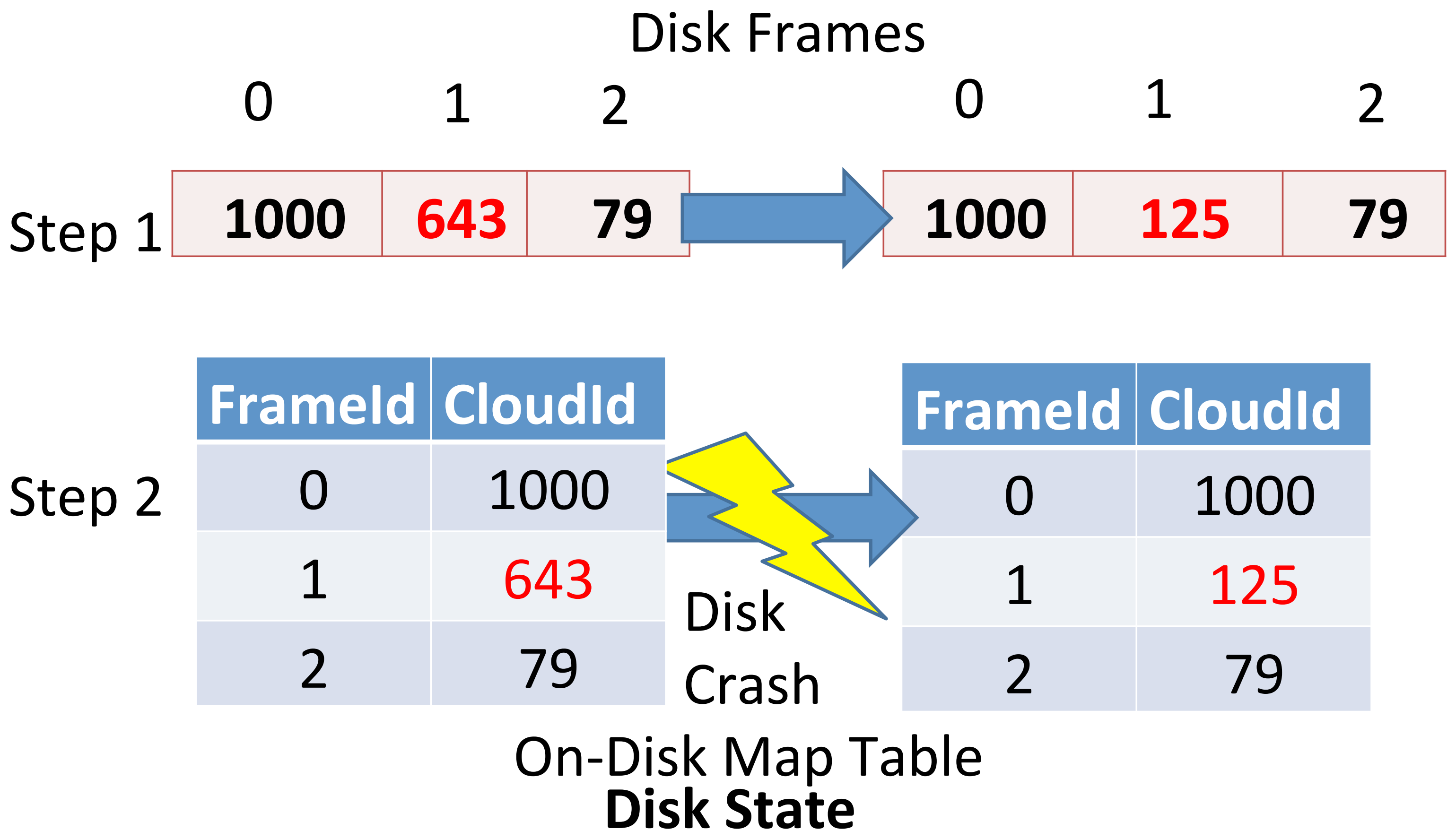
System Architecture



System Components
- **Device Driver**(**Cloud Cache)** responsible for serving read/write request from page cache.
- The Driver manages the Disk as a cache and requests Cloud Client Process for eviction/ installation requests .
- **Cloud Client Process** - responsible for eviction/installation from Cloud to the disk and writeback of dirty data.

Making Block-level Storage Smarter

Disk Cache Consistency Across Crashes



Disk Frames

Step 1

0	1	2
1000	643	79

0	1	2
1000	125	79

Step 2

FrameId	CloudId
0	1000
1	643
2	79

Disk Crash

FrameId	CloudId
0	1000
1	125
2	79

On-Disk Map Table

Disk State

Consistency:
- Transactional updates to disk frames and on-disk map-table are required.
- **Updates require extra disk seek per first write to a disk frame**
- **Have** huge cost penalty.
- Use journal commit block and super block update to checkpoint map-table entry improving performance cost.

Performance - Fast Writes

File System writes do not require Cloud Blocks to be installed on Disk for execution.
- An optimization to provide Disk Like Latencies for writes.
- Pose concurrency challenges with transactional update to disk frame and suggests **check-pointing** as a smart method to ensure disk cache consistency.

Capacity - Deletes

File Systems like ext3 assumes the storage device to be a disk.
- In ext3, data liveness is inferred by block bitmaps.
- Files are deleted and data blocks are released,
- Bitmaps are reset
- The data still sits on the Disk.
- Release the data blocks which are no more in use by the File System.
- **Snoop journal =>** infer liveness[2] at Block Level
- In Future, implement the trim command for explicitly deleting data

Evaluation

Microbenchmark

Reads
- Disk-like Performance for cached Data.
- Cloud-like Performance for uncached data.

Writes
- Disk-like performance **always!!!**

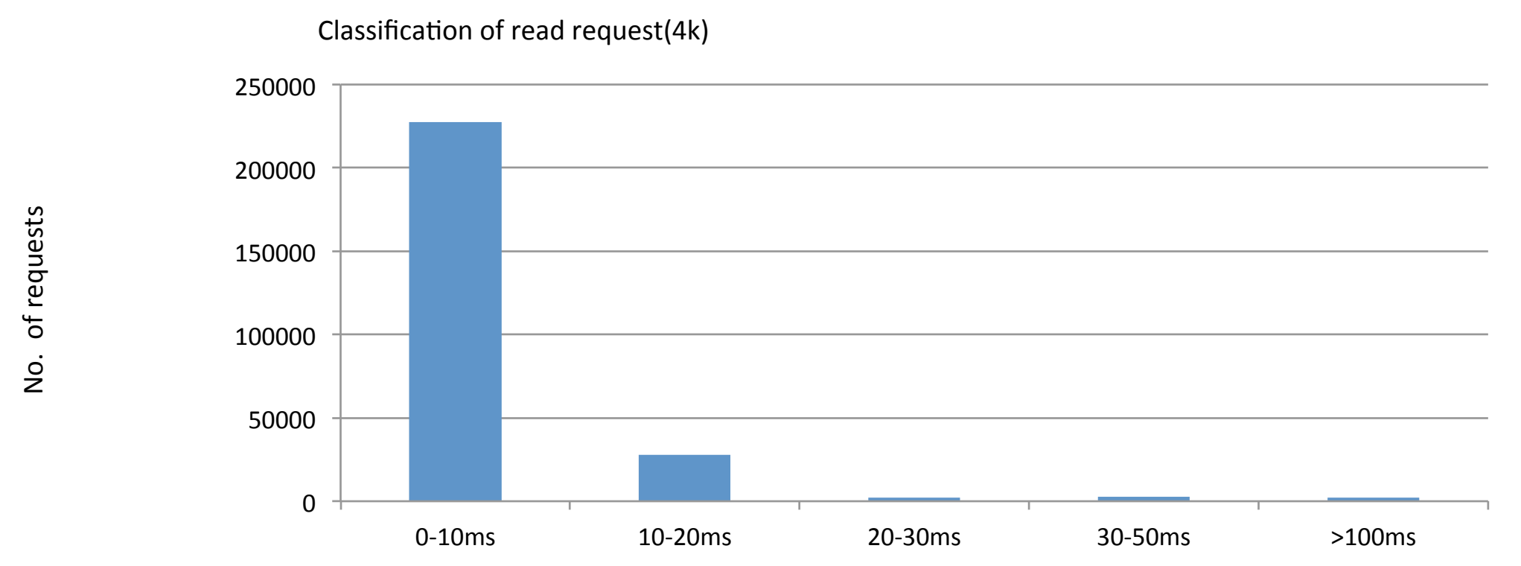
Disk Cache Size: 16G
Cloud Storage Size : 128G

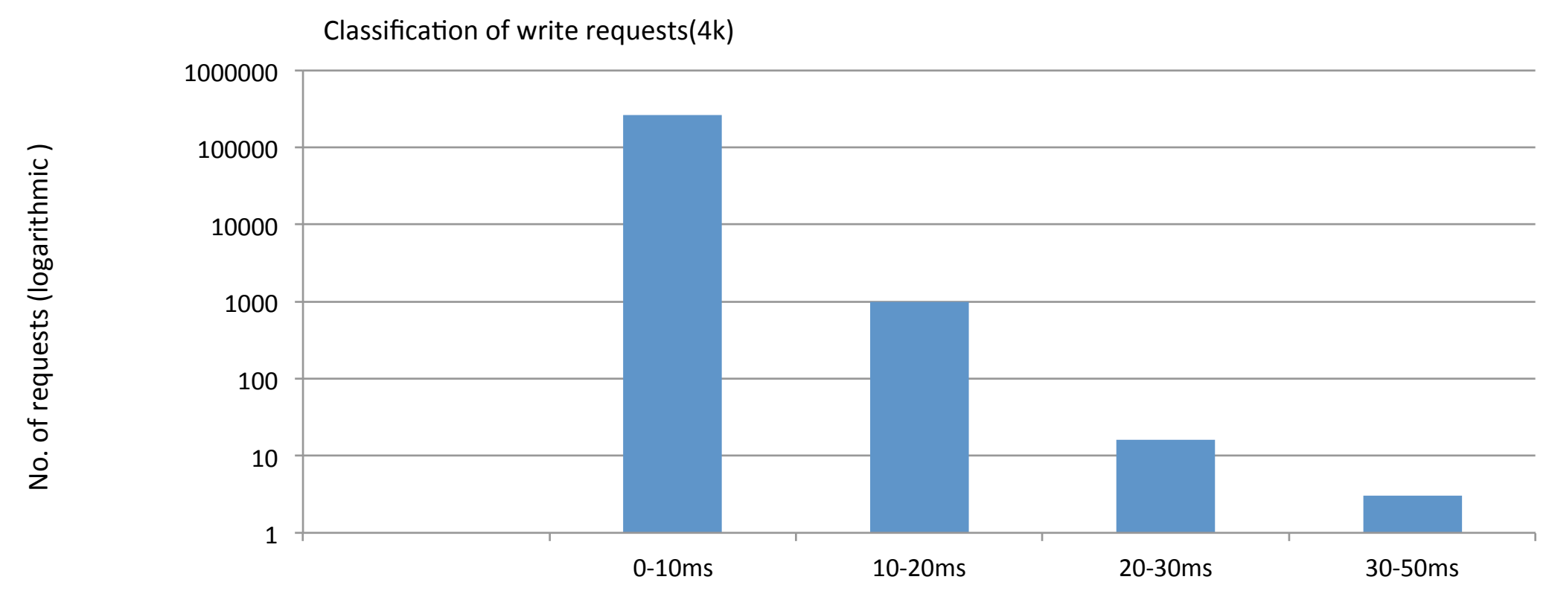
Cache Consistency Performance:

	Hot Cache	Cold Cache
Random Read	6.19ms	1713ms/10.03ms
Random Write	8.14ms	2904ms/12.17ms
Sequential Read	39.3MB/s	3.89MB/s
Sequential Write	32.8MB/s	3.06MB/s

Fast Writes with Transactional Update		
Random Write	6.69ms	**31.89ms/22.75ms**
Fast Writes with Journal Guided Checkpointing		
Random Write	6.69ms	**21.34ms/8.87ms**

Raw Disk		
Random Read	6.00ms	8.36ms
Random Write	7.54ms	9.28ms
Sequential Read	40MB/s	40MB/s
Sequential Write	32.3MB/s	36.7MB/s





References and Related Work

[1] V. Chidambaram, T. Sharma, Andrea C. Arpaci-Dusseau and Remzi H. Arpaci-Dusseau: Consistency With Ordering. In *FAST '12.*

[2] M. Sivathanu, L. Bairavasundaram, Andrea C. Arpaci-Dusseau and Remzi H. Arpaci-Dusseau: Life or Death at Block-Level In *OSDI' 04.*

[3] T. Denehy, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau: Journal-guided Resynchronization for Software RAID in *FAST'05*.

[4] M. Vrable, S. Savage, and G. M. Voelker: Cumulus: File System Backup to the Cloud in *TOS'09.*