Cumulus: Filesystem Backup to the Cloud

7th USENIX Conference on File and Storage Technologies (FAST '09)

Michael Vrable Stefan Savage Geoffrey M. Voelker

University of California, San Diego

February 26, 2009

Introduction

- Cloud computing important emerging area, with a spectrum of implementations
- "Thick" cloud: Purchase a complete integrated service from a provider
  - Potentially greater efficiencies
  - Easier to set up

- "Thin" cloud: Customer builds application on more generic services
  - More choices among service providers
  - Easier to migrate between providers
  - Potentially lower costs

- Thin cloud offers some advantages, particularly for applications such as backup
  - How well can we do with such a simple interface?

Cumulus: Background and Requirements

- Network Backup: Functionality
  - Implement backup over a network to provide easy off-site storage
  - Store snapshots of file data at multiple points in time
  - Allow recovery of selected files or entire snapshot

Cumulus: Background and Requirements

- Network Backup: Functionality
  - Implement backup over a network to provide easy off-site storage
  - Store snapshots of file data at multiple points in time
  - Allow recovery of selected files or entire snapshot
- System Requirements
  - Build on a thin cloud model: simple storage interface only
  - Storage layer need only support `put`/`get` of blobs of data, `list`, `delete`
  - Implies that application logic must be built into client
  - Focus on cloud storage, but could be FTP server, friend's computer, P2P network, ...

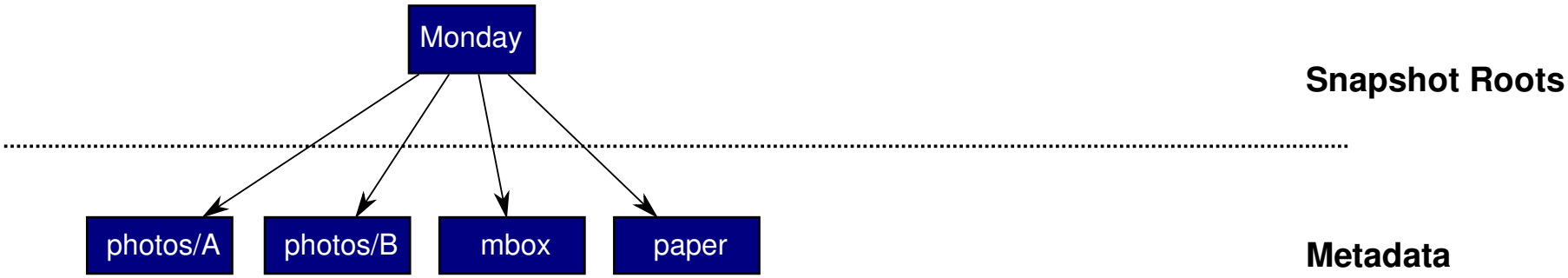
Cumulus: Background and Requirements

- Network Backup: Functionality
  - Implement backup over a network to provide easy off-site storage
  - Store snapshots of file data at multiple points in time
  - Allow recovery of selected files or entire snapshot
- System Requirements
  - Build on a thin cloud model: simple storage interface only
  - Storage layer need only support `put`/`get` of blobs of data, `list`, `delete`
  - Implies that application logic must be built into client
  - Focus on cloud storage, but could be FTP server, friend's computer, P2P network, ...
- Goals
  - Minimize resource requirements (storage, network)
  - Minimize ongoing monetary costs

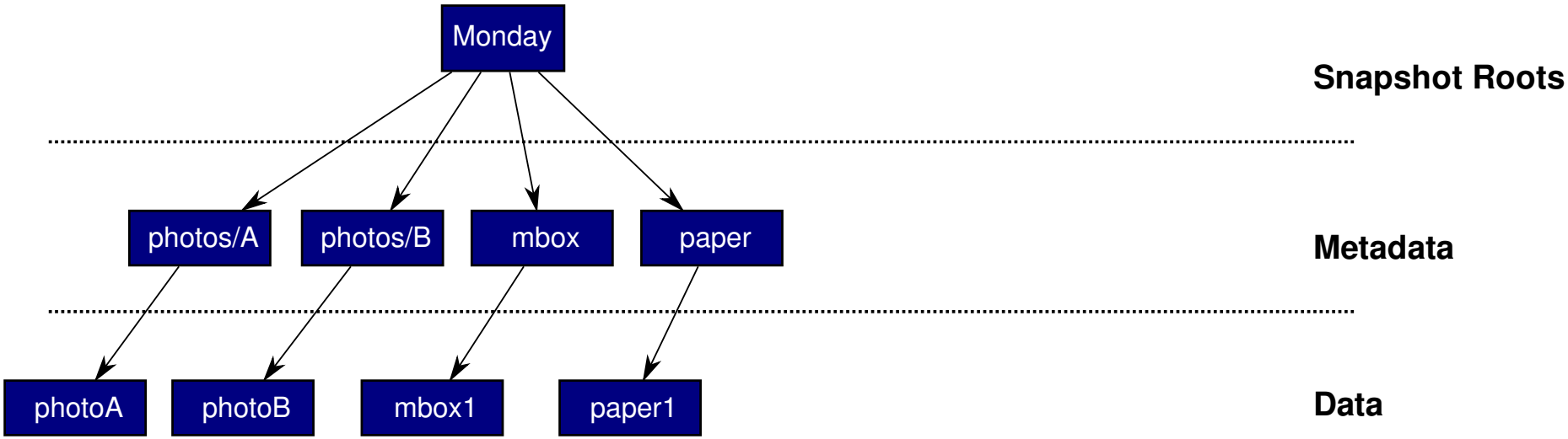
Cumulus Backup Format

Snapshot Roots

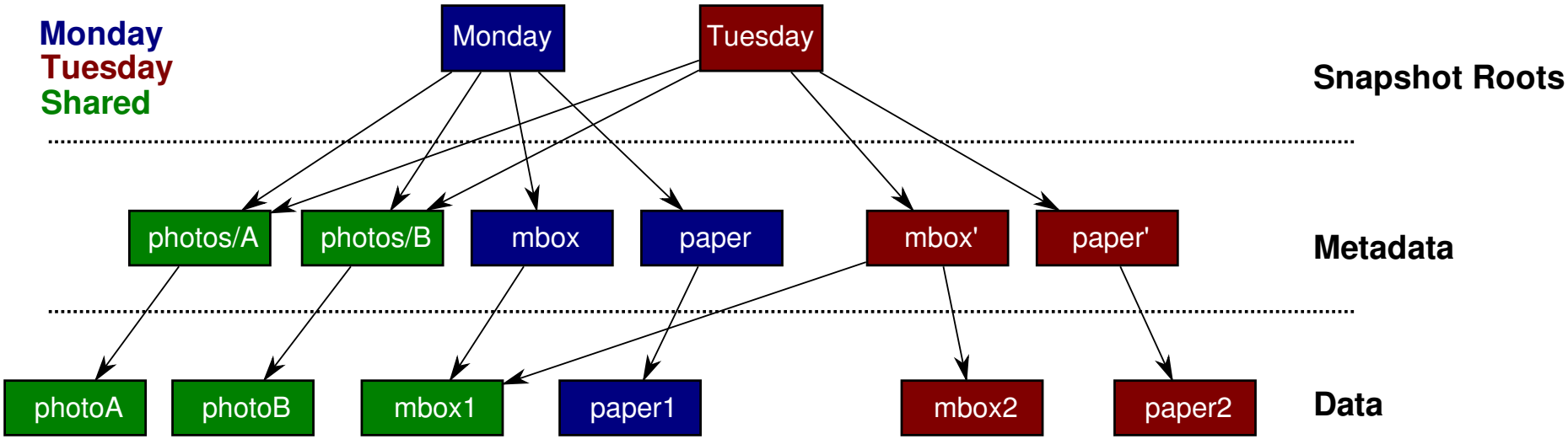
Cumulus Backup Format



Cumulus Backup Format

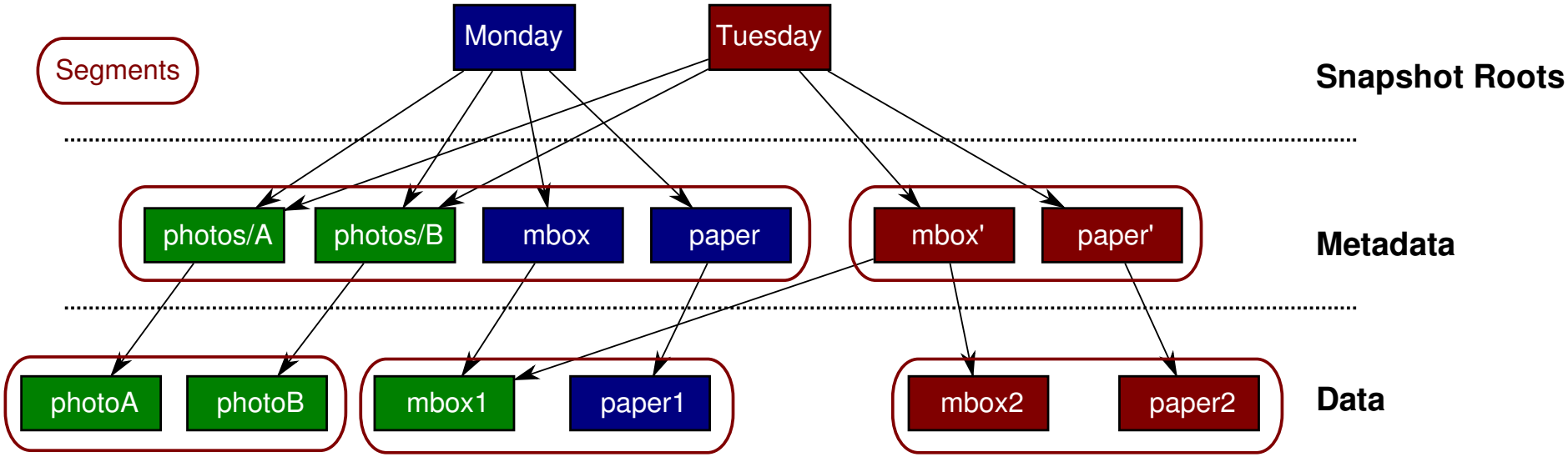


Cumulus Backup Format



- Stores filesystem snapshots at multiple points in time
- Data blocks shared within, between snapshots
- Minimizes storage, upload bandwidth needed

Aggregation: Minimizing Per-Block Costs

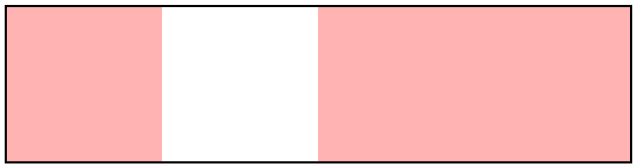


- May have per-file in addition to per-byte costs
  - Protocol overhead: Slower backups from more transactions
  - Per-file overhead at storage server
  - May be exposed as monetary cost by provider
- Cumulus reduces these costs by aggregating blocks into **segments** before storage
  - Aggregation follows from our constraints, but may not be needed in other systems

Aggregation Challenges: Internal Fragmentation

Day 1

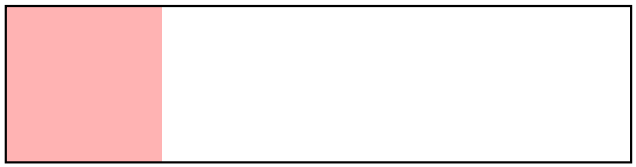
Aggregation Challenges: Internal Fragmentation



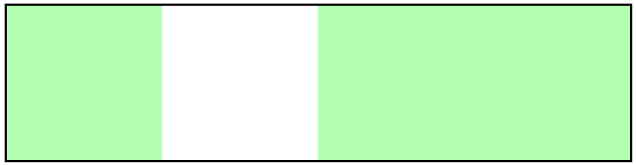
Day 1

Day 2

Aggregation Challenges: Internal Fragmentation



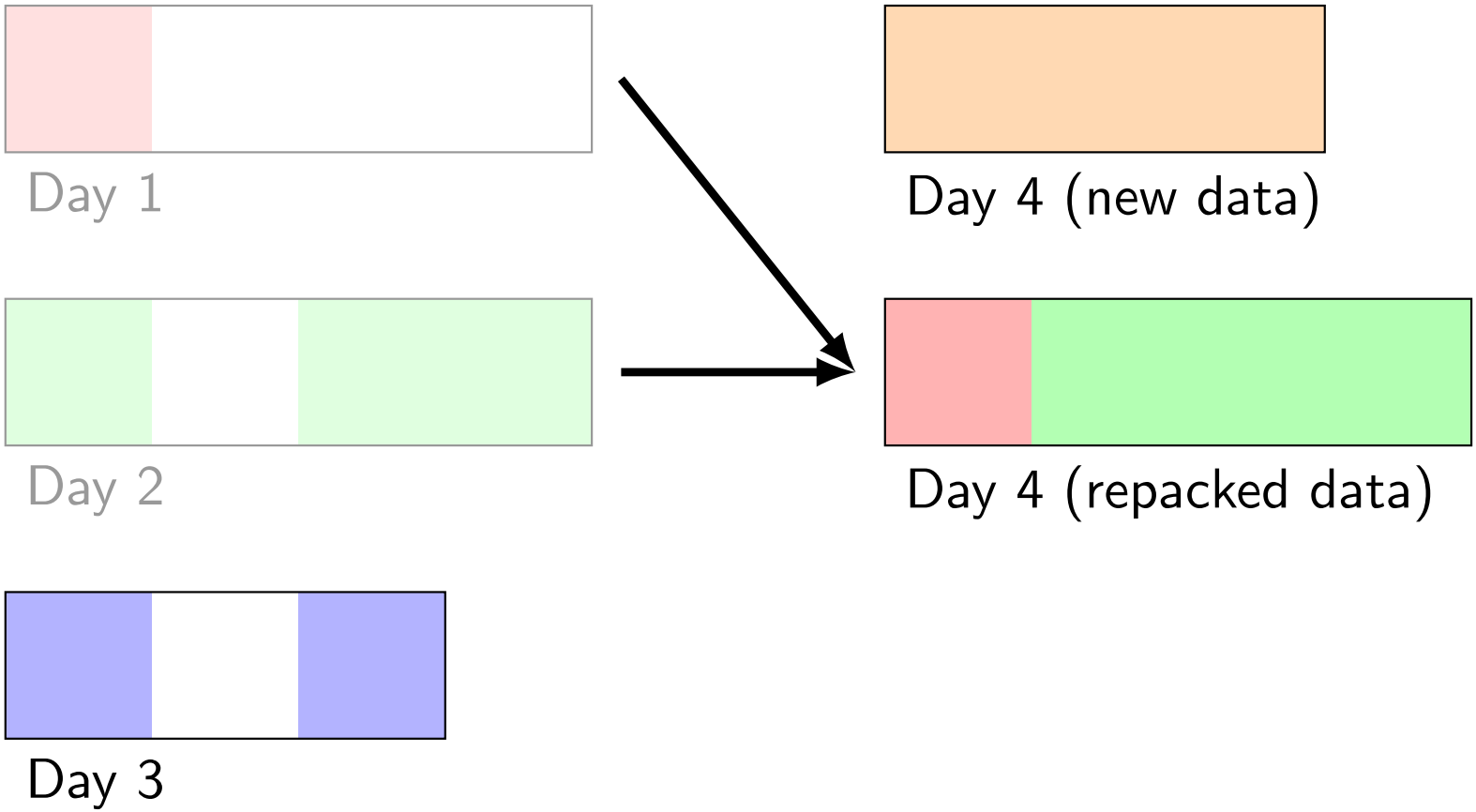
Day 1



Day 2

Day 3

Aggregation Challenges: Internal Fragmentation



- Wasted space within segments reclaimed by **segment cleaning**
- Tradeoff: space vs. upload bandwidth
- Contribution: Show how to tune segment size, threshold for cleaning

Cumulus Implementation

- Implemented as $\approx$ 4000 lines C++, Python
- Execution packages new data into segments, uploads to storage server
- Client tracks some data locally (not essential for restores):
  - Block hash database
  - Previous snapshot metadata (detect changed files)
- Other features:
  - Compression/encryption
  - Sub-file incremental updates
- More details in the paper

- In real use: I have been using it for over 18 months

Evaluation

Key Questions:

- What is the resource (network, storage) overhead imposed by the restricted storage interface?
- How do these overheads translate into monetary terms?
- How can aggregation and cleaning be tuned to minimize the cost?
- How does the prototype perform?

Evaluation Traces

	Fileserver	**User**
Duration (days)	157	223
Entries	26673083	122007
Files	24344167	116426
File Sizes		
Median	0.996 KB	4.4 KB
Average	153 KB	21.4 KB
Maximum	54.1 GB	169 MB
Total	3.47 TB	2.37 GB
Update Rates		
New data/day	9.50 GB	10.3 MB
Changed data/day	805 MB	29.9 MB
Total data/day	10.3 GB	40.2 MB

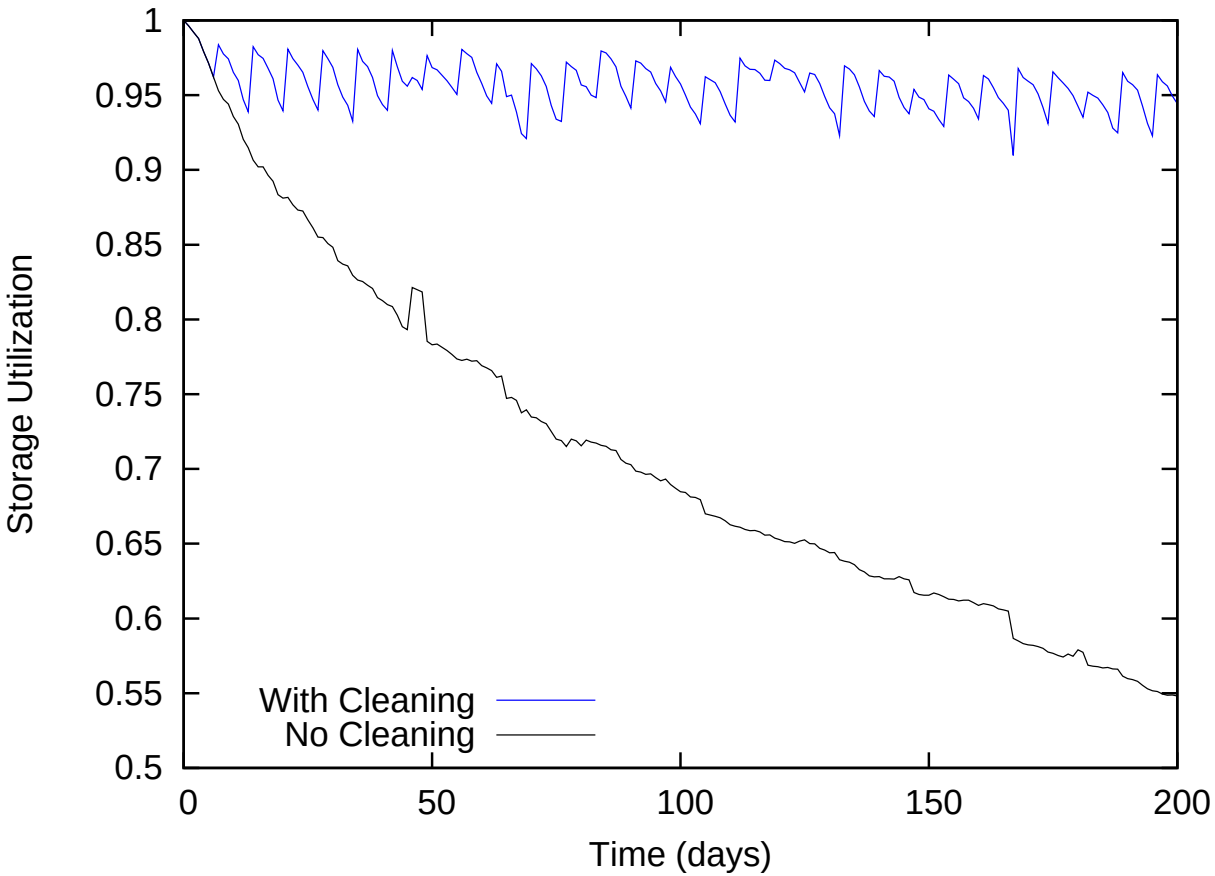
Evaluation Traces

	Fileserver	User
Duration (days)	157	223
Entries	26673083	122007
Files	24344167	116426
File Sizes		
Median	0.996 KB	4.4 KB
Average	153 KB	21.4 KB
Maximum	54.1 GB	169 MB
Total	3.47 TB	2.37 GB
Update Rates		
New data/day	9.50 GB	10.3 MB
Changed data/day	805 MB	29.9 MB
Total data/day	10.3 GB	40.2 MB

Backup Simulation

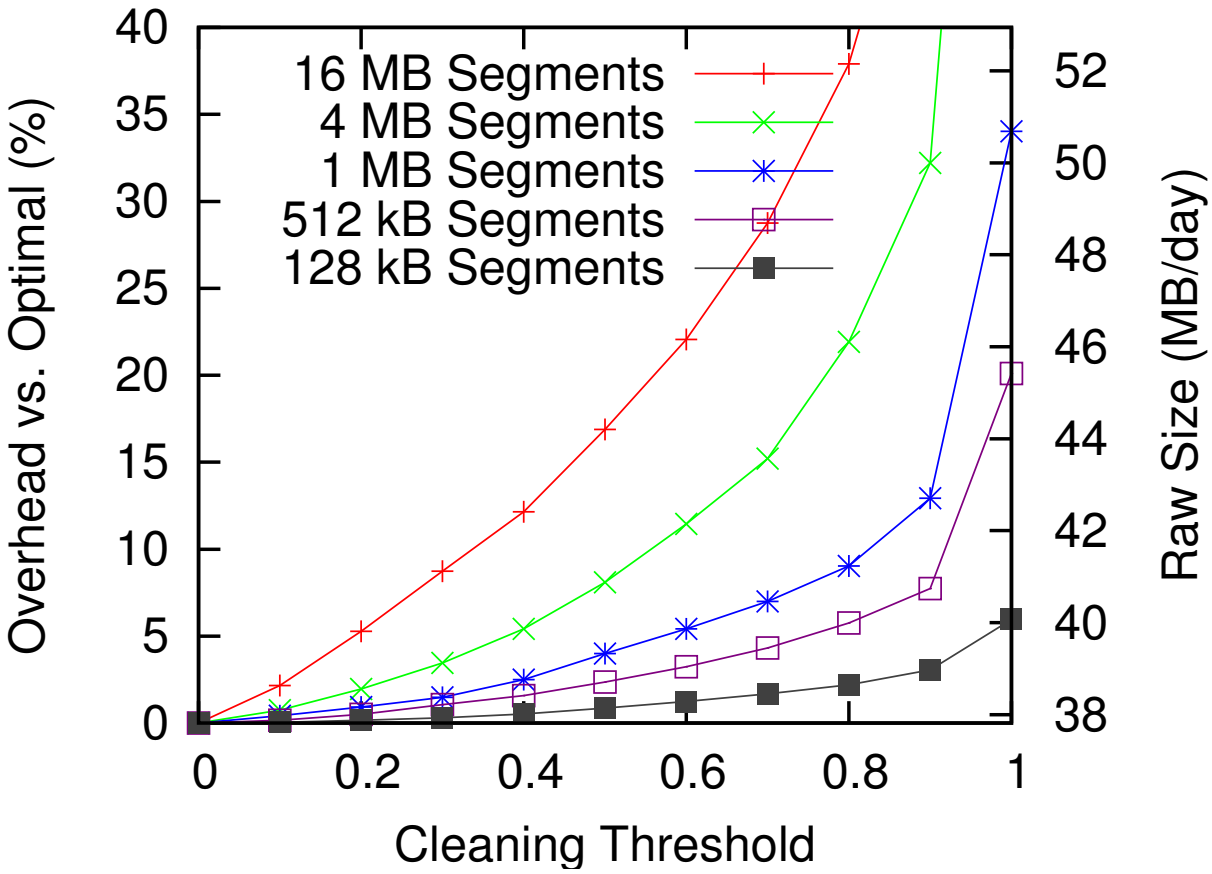
- Compare against **optimal** backup performance:
  - All unique data *must* be stored at server
  - All new data *must* be transferred over network
- In simulation, compare Cumulus against these baseline values
- Consider effect of aggregation, cleaning parameters
- For simplicity, ignore compression and metadata
  - Effects discussed in paper

Is Cleaning Necessary?



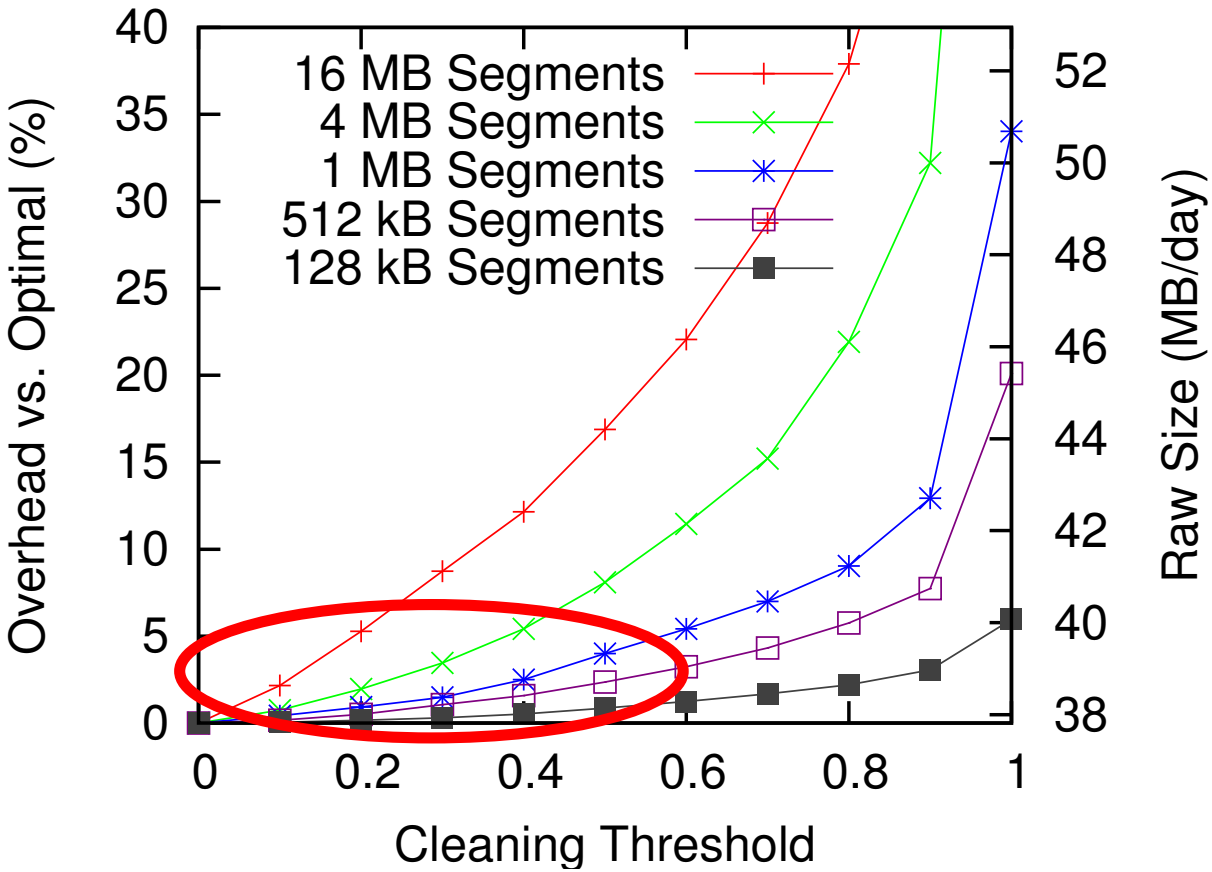
- Without segment cleaning, storage utilization steadily decreases
- Weekly cleaning keeps overhead within a narrow range
- Exact overhead depends on cleaning parameters

How Much Data is Transferred?



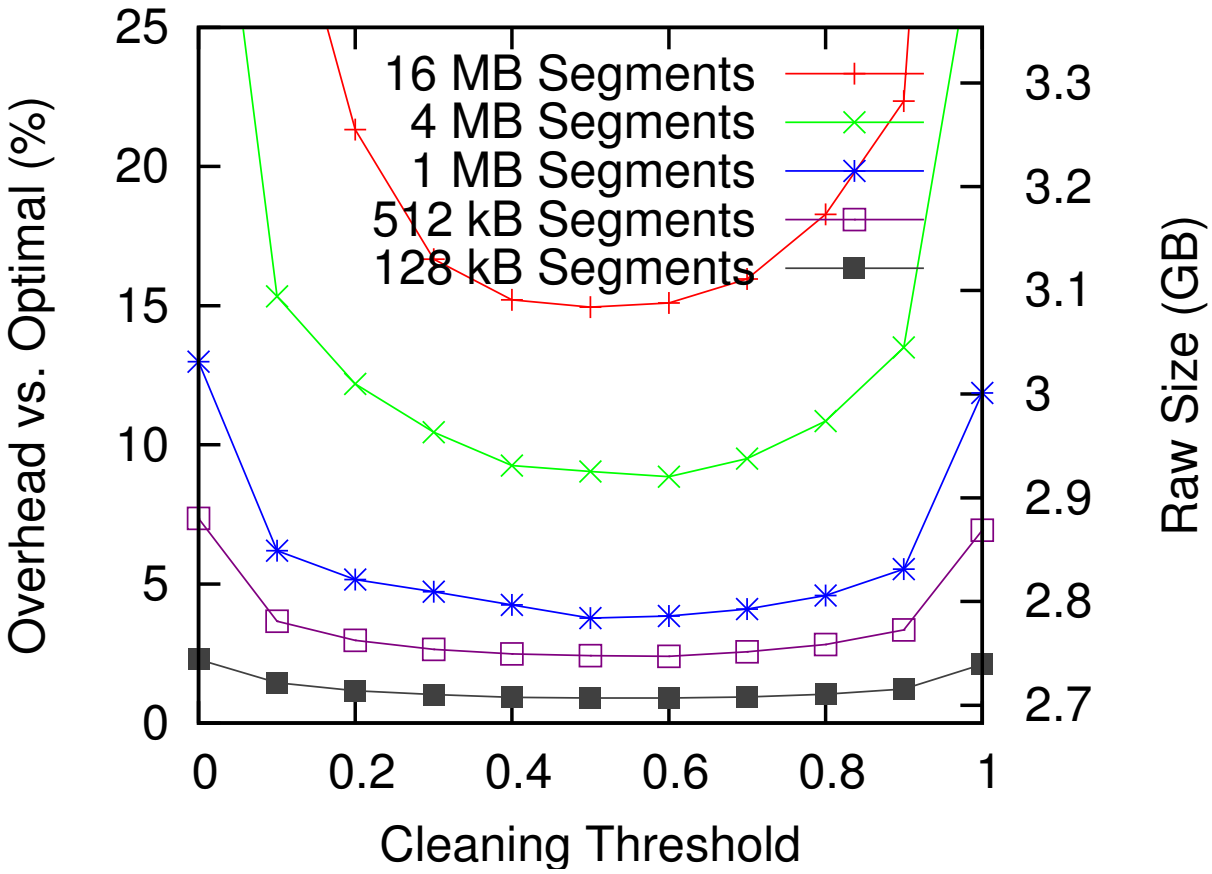
- Aggressive cleaning, large segments increase overhead

How Much Data is Transferred?



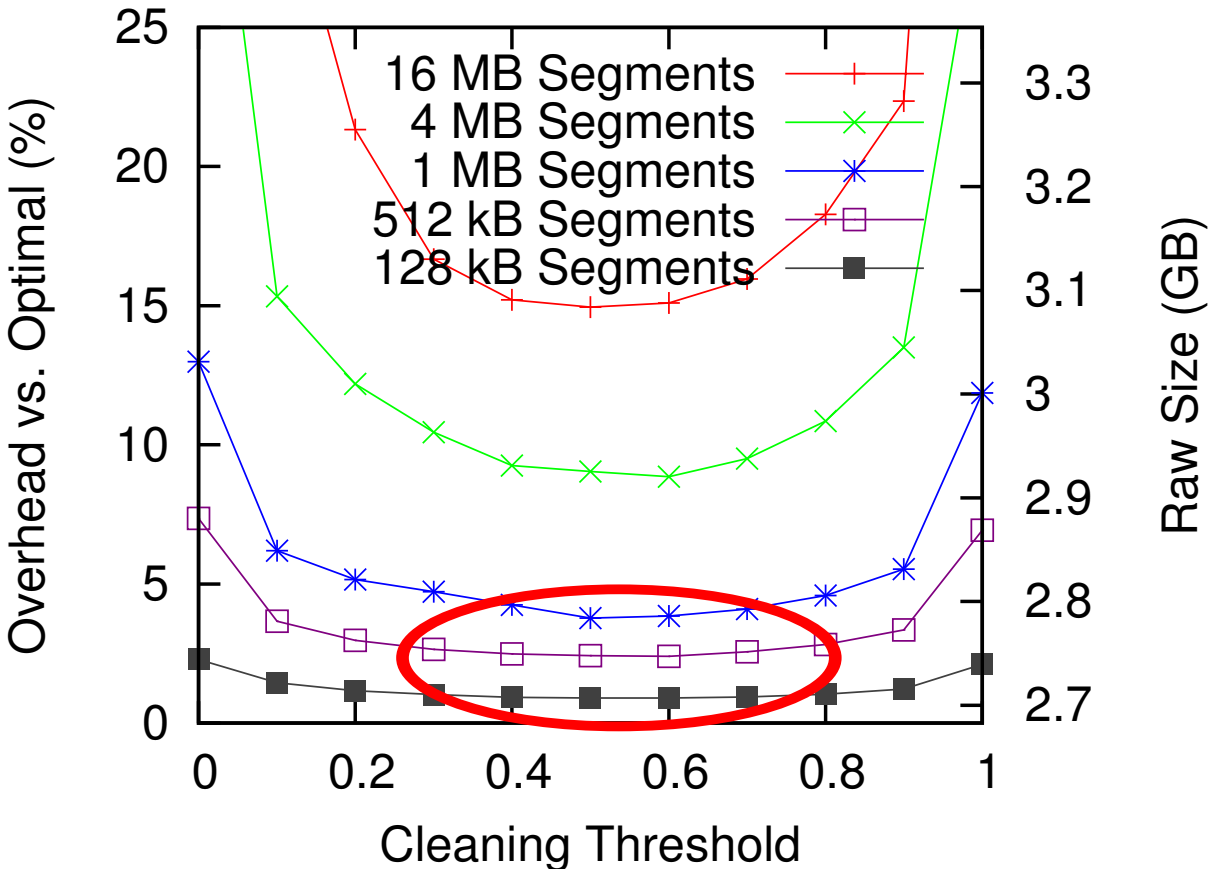
- Aggressive cleaning, large segments increase overhead

What is the Storage Overhead?



- Large segments increase overhead
- Too little cleaning leads to large overheads
- Aggressive cleaning leads to churn, storage overhead when keeping multiple snapshots

What is the Storage Overhead?



- Large segments increase overhead
- Too little cleaning leads to large overheads
- Aggressive cleaning leads to churn, storage overhead when keeping multiple snapshots

Estimating Ongoing Backup Costs

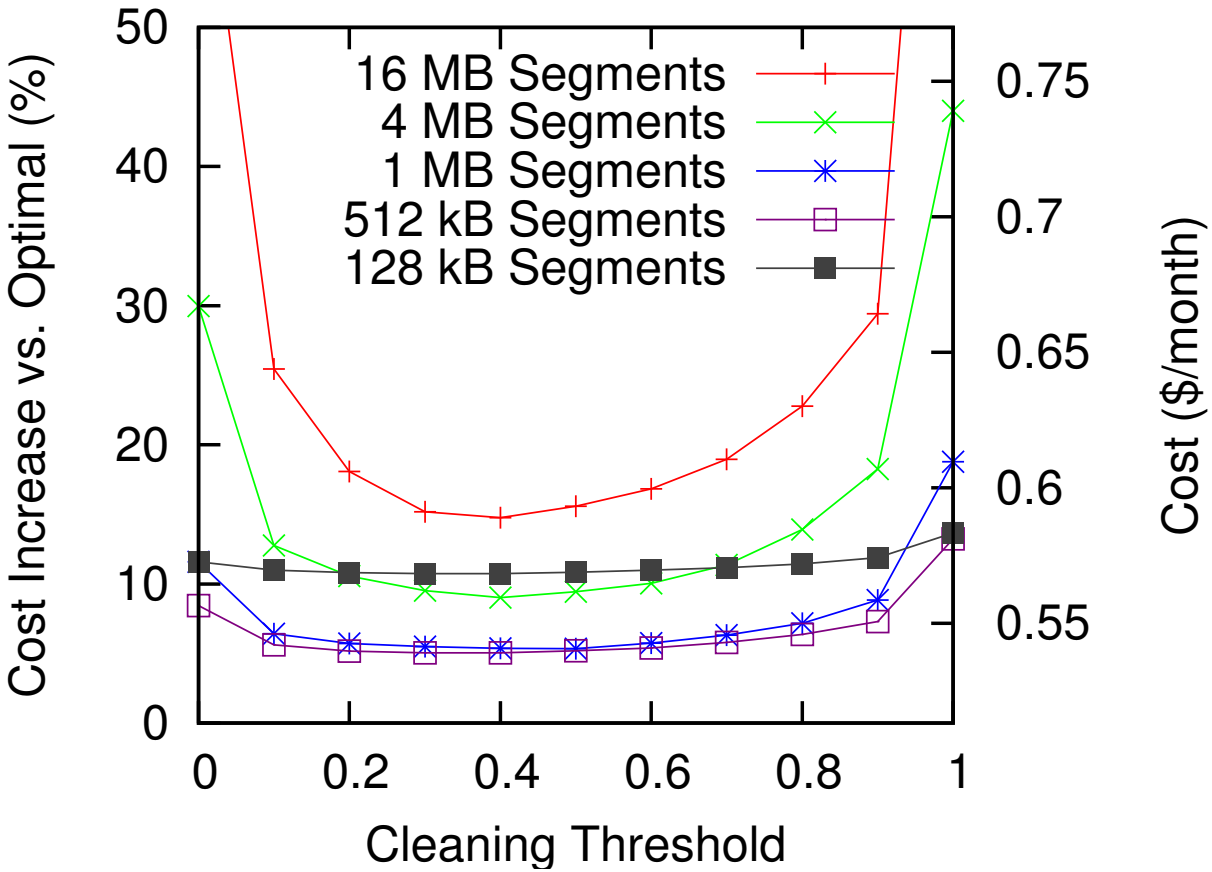
- How do storage, upload translate into total **cost** for implementing backup?
- Amazon S3 prices:

  Storage: \$0.15 per GB · month

  Upload: \$0.10 per GB

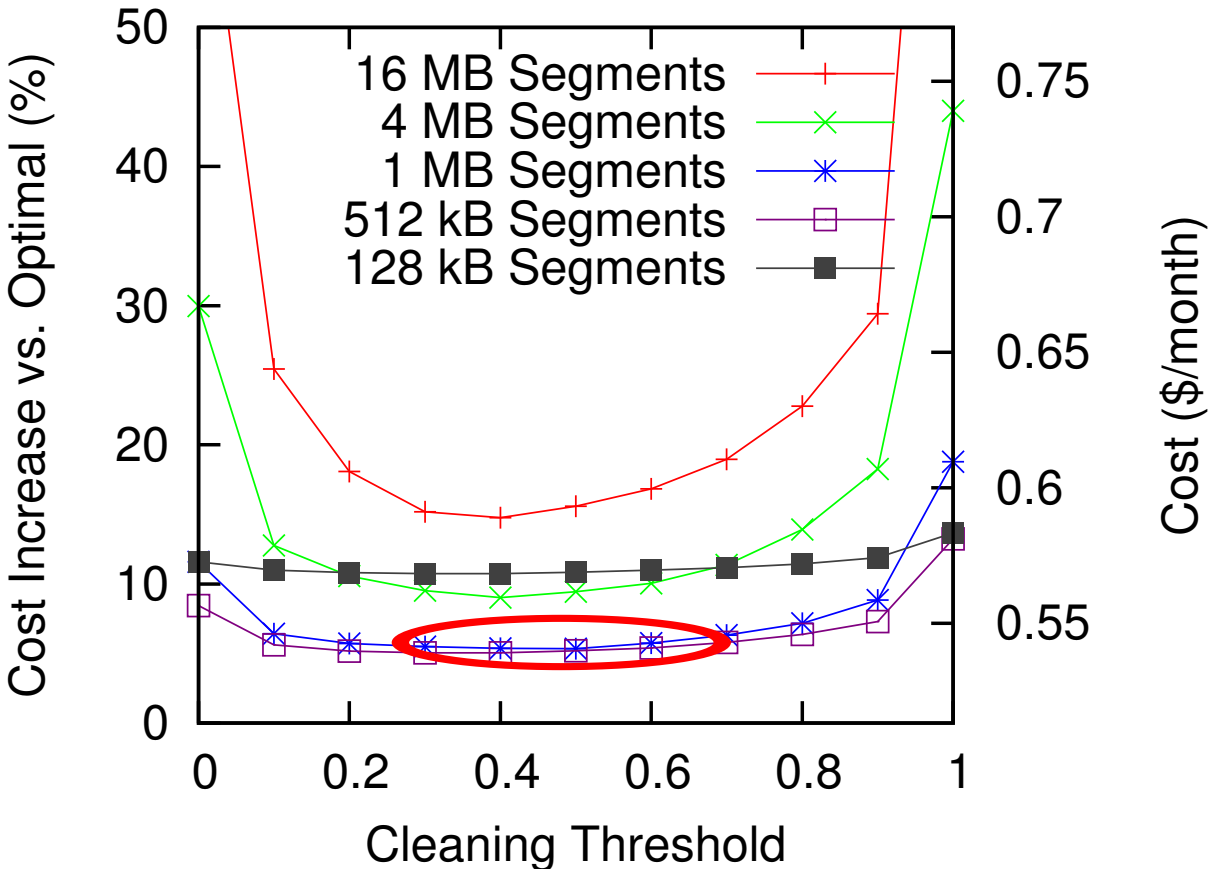
  Operation: \$0.01 per 1000 uploads
- Effects of varying costs discussed in the paper

What Settings Minimize Total Cost?



- Aggressive cleaning, large segments increase overhead
- Total cost includes per-segment charge: intermediate segment size is best
- Cleaning threshold 0.4–0.6, segment size 0.5–1 MB work well

What Settings Minimize Total Cost?



- Aggressive cleaning, large segments increase overhead
- Total cost includes per-segment charge: intermediate segment size is best
- Cleaning threshold 0.4–0.6, segment size 0.5–1 MB work well

Simulation Summary

- Storage cost dominates ($> 75\%$ in this trace)
- Cost not overly sensitive to aggregation, cleaning settings
- Cost within 5–10% of best we could expect
  - Implications for integrated backup?

Prototype Evaluations

- Tested full prototype using backups from two months of user trace
  - Snapshots stored properly, could be restored
  - Ongoing costs come out to $0.24/month for around 2 GB of data
- Compared with two existing tools for Amazon S3
  - Brackup and JungleDisk: two other tools capable of filesystem backup to S3
  - Monthly costs are 19–200% more
  - But, systems designed for more than just backup or not explicitly tuned for cost
- What about thick cloud?
  - Mozy: integrated online backup solution
  - $5/month for "unlimited" backups
  - $0.50/GB/month for businesses

Summary

- Cumulus is a cost-effective tool for backup to network storage
- We show how system parameters can be tuned to minimize total cost
- Shows specialized server not necessary for implementing low-overhead backup
  - Can choose from variety of storage providers based on cost or other factors

Questions?

Cumulus is available at
`http://sysnet.ucsd.edu/projects/cumulus/`

Deduplication

- Cumulus implementation does perform coarse-grained data deduplication
  - Recognizes duplicate data at file or 1 MB block level
  - Block boundaries for deduplication are fixed
- Deduplication only for a single client, not across clients
- Server-side support could enable deduplication across clients
  - Doesn't work well with aggregation into segments
  - Does slightly reduce privacy of backup
  - Complicates accounting