Delta Compressed and Deduplicated Storage Using Stream-Informed Locality

Philip Shilane, Grant Wallace, Mark Huang, & Windsor Hsu

Backup Recovery Systems Division
EMC Corporation

Motivation and Approach

- Improve storage compression
  - Decrease price per GB
  - Decrease data center space
  - Decrease power
  - Decrease management
- Combine deduplication and delta compression
  - Remove identical data regions
  - Compress with similar data regions

EMC²

Previous Work on Similarity Indexing

- Version information [Burns’97, MacDonald’00]
- Similarity index in memory [Aronovich’09, Kulkarni’04]
- Similarity index on-disk [You’11]
- Stream-informed delta locality for WAN replication [Shilane’12]
  - Low WAN throughput
  - Did not store delta compressed data

EMC²

Contributions

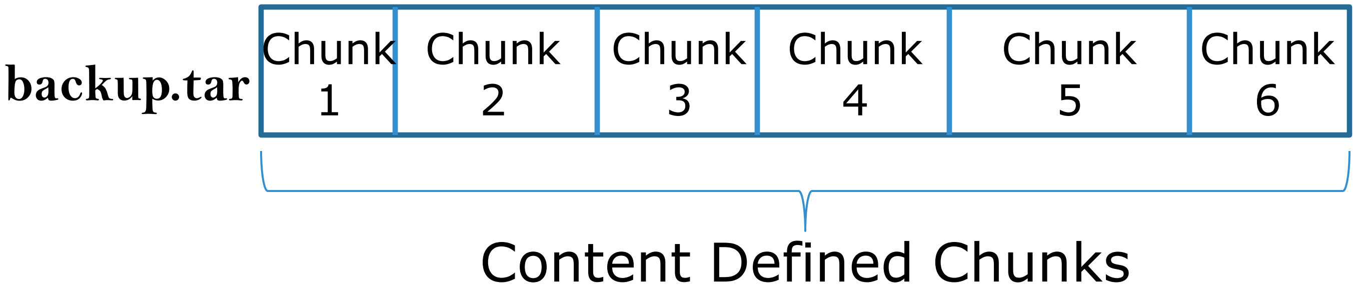
1. First deduplicated and delta compressed storage implementation using stream-informed locality
2. Quantify throughput and suggest improvement areas
3. Explore new complexities related to data integrity and cleaning
4. Report combination of deduplication and delta compression across chunk sizes

EMC²

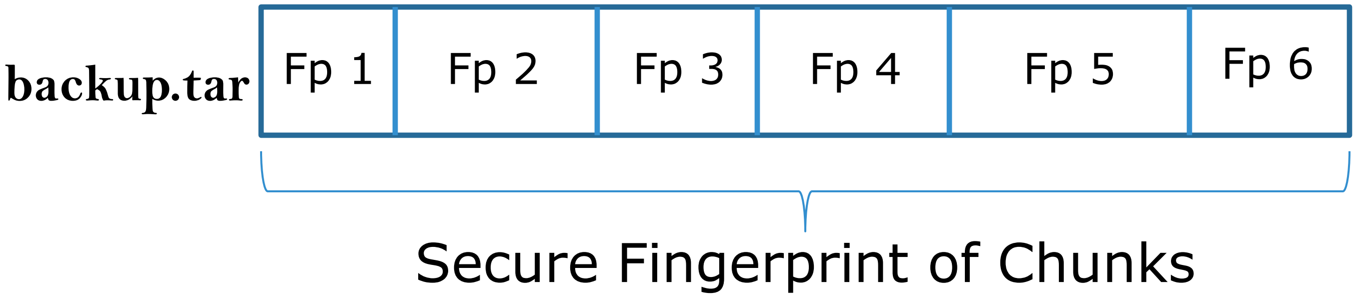
Stream-informed Deduplication

backup.tar

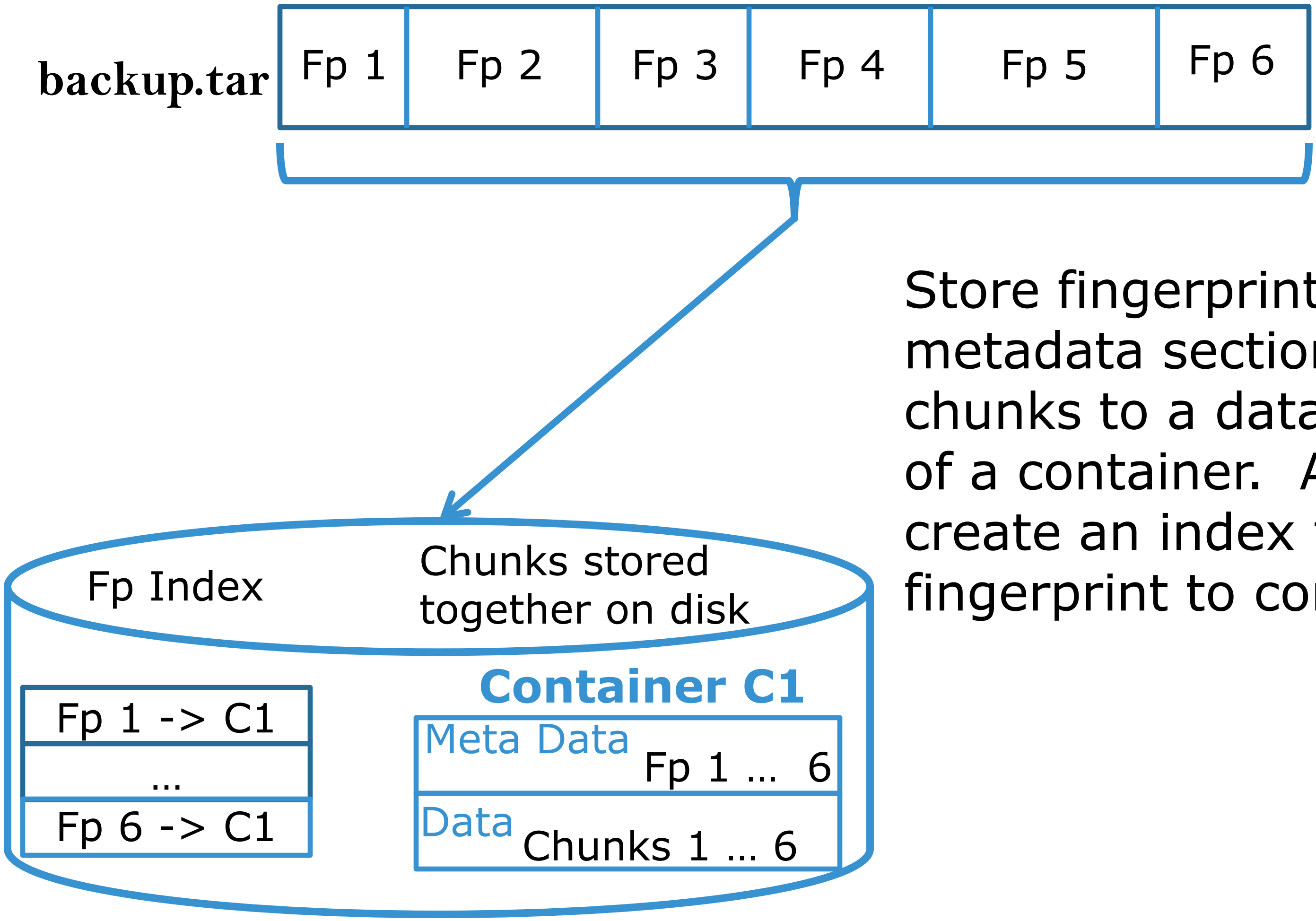
Stream-informed Deduplication



Stream-informed Deduplication

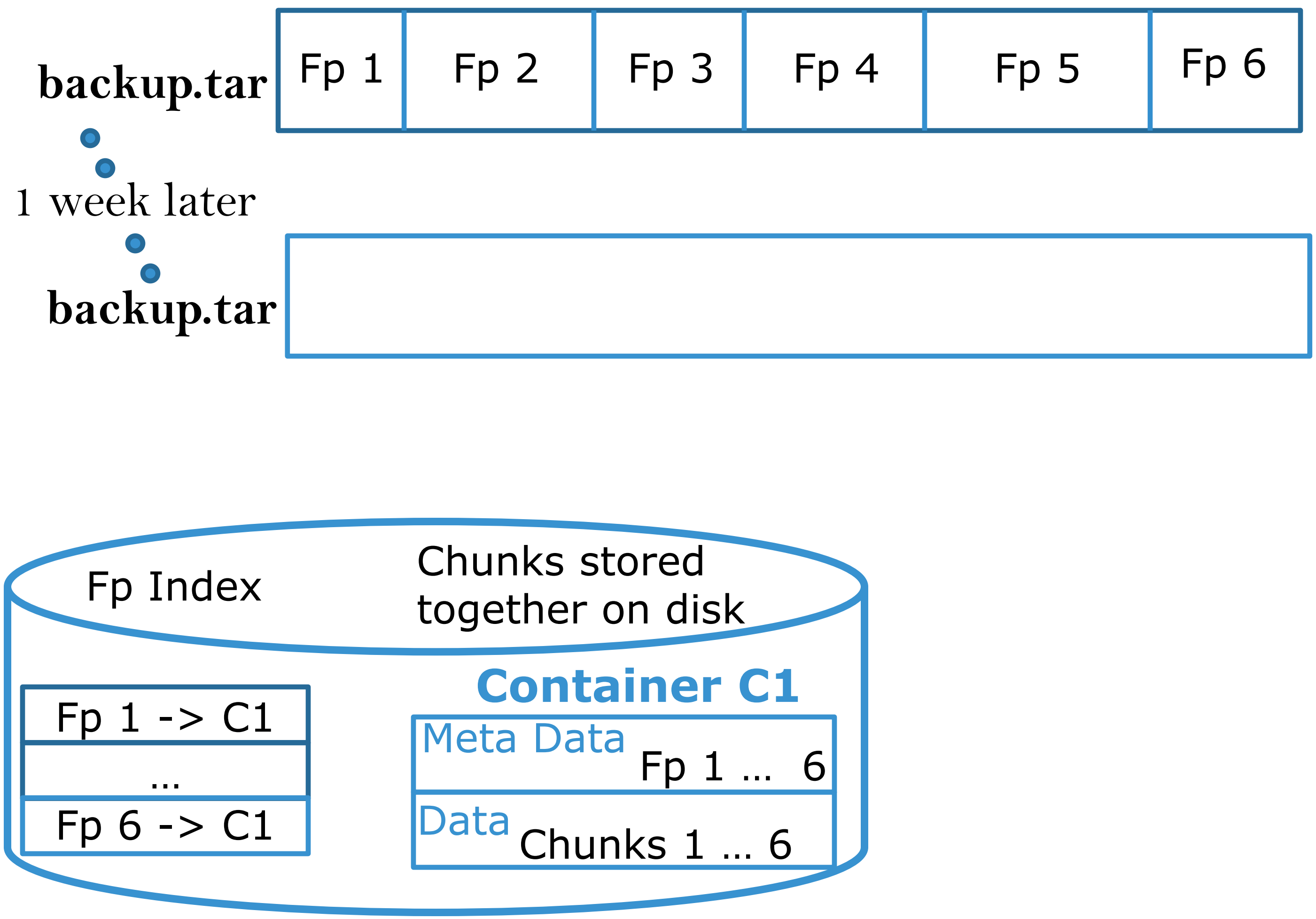


Stream-informed Deduplication

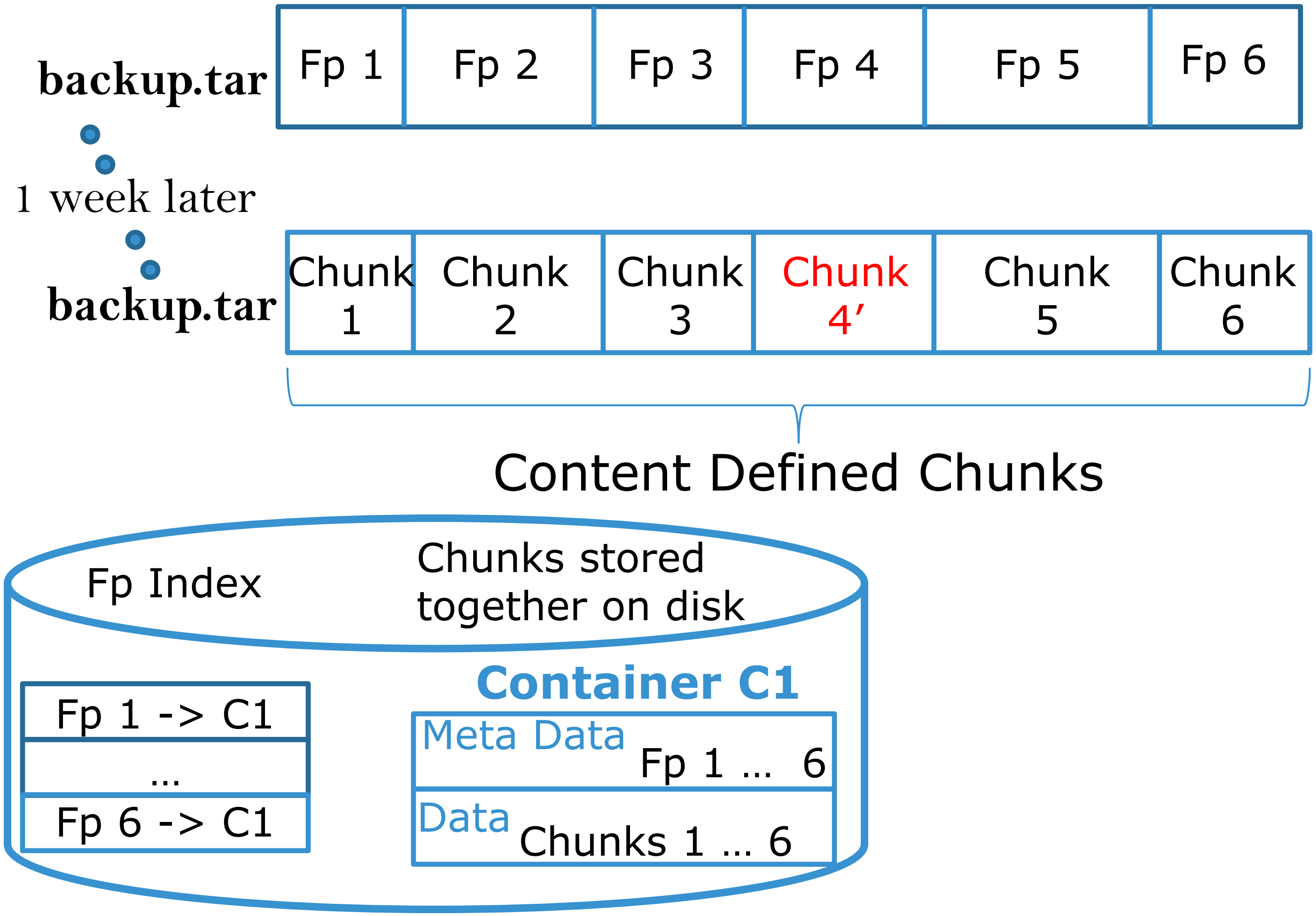


Store fingerprints in a metadata section and chunks to a data section of a container. Also create an index from fingerprint to container.

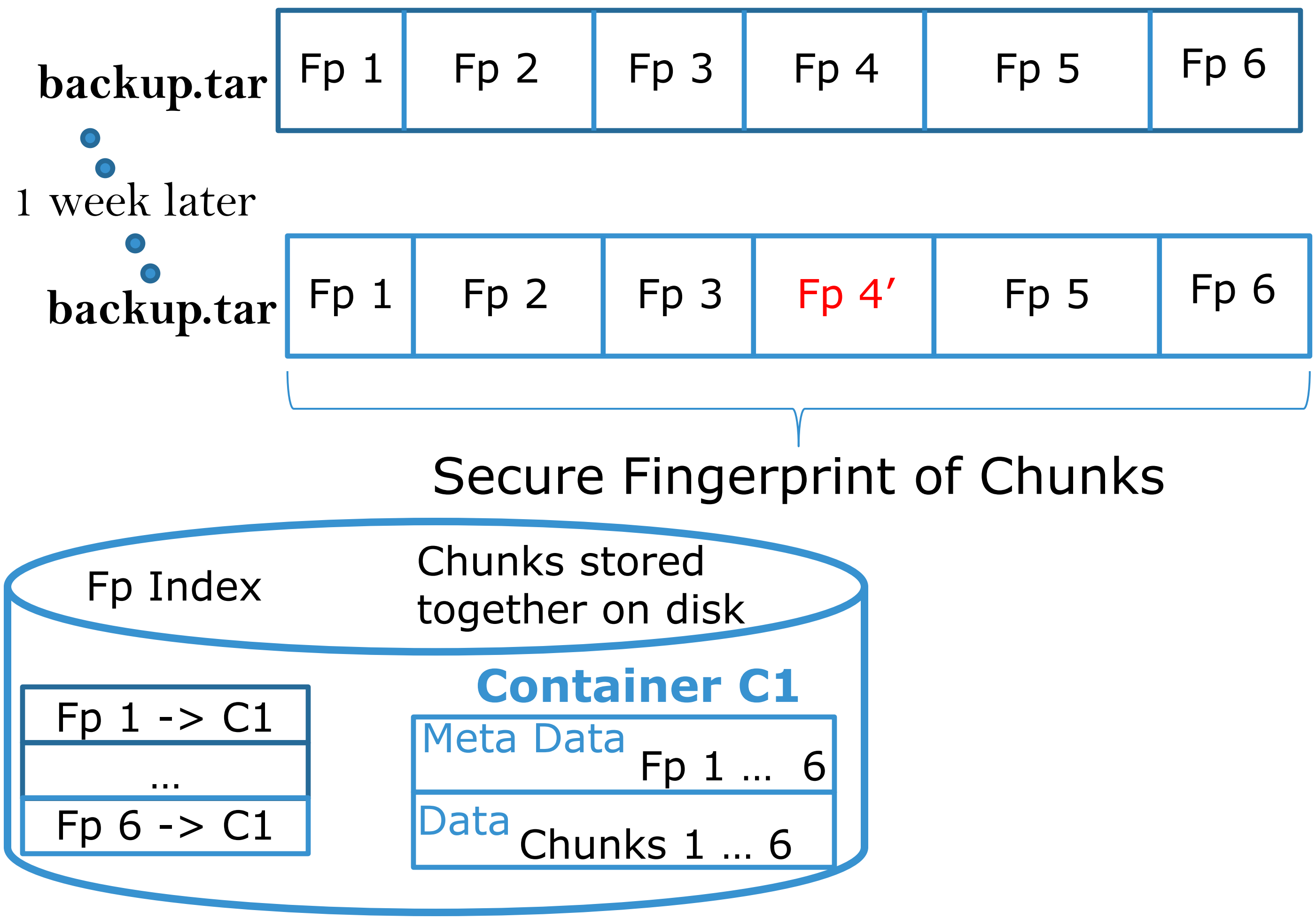
Stream-informed Deduplication



Stream-informed Deduplication

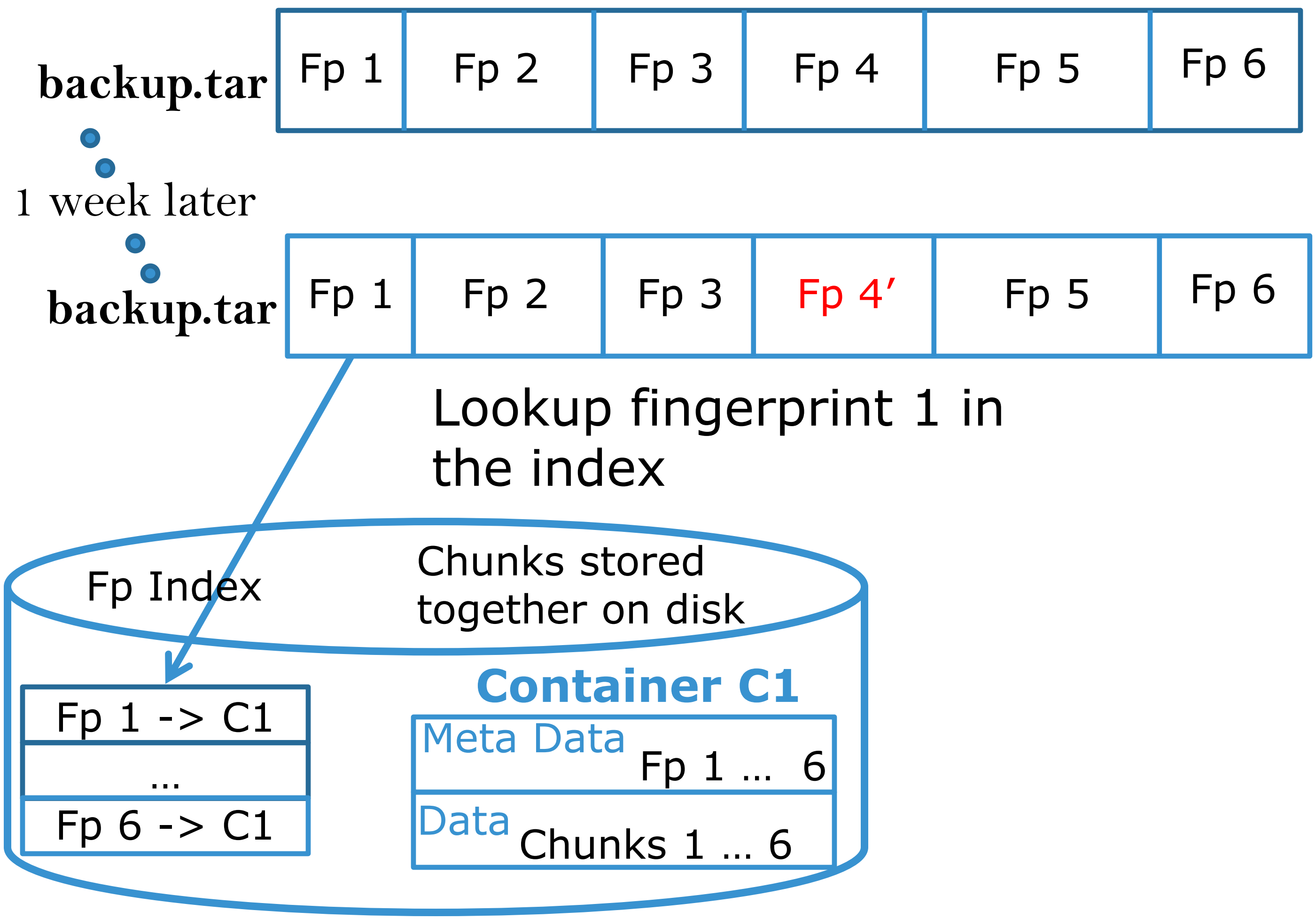


Stream-informed Deduplication

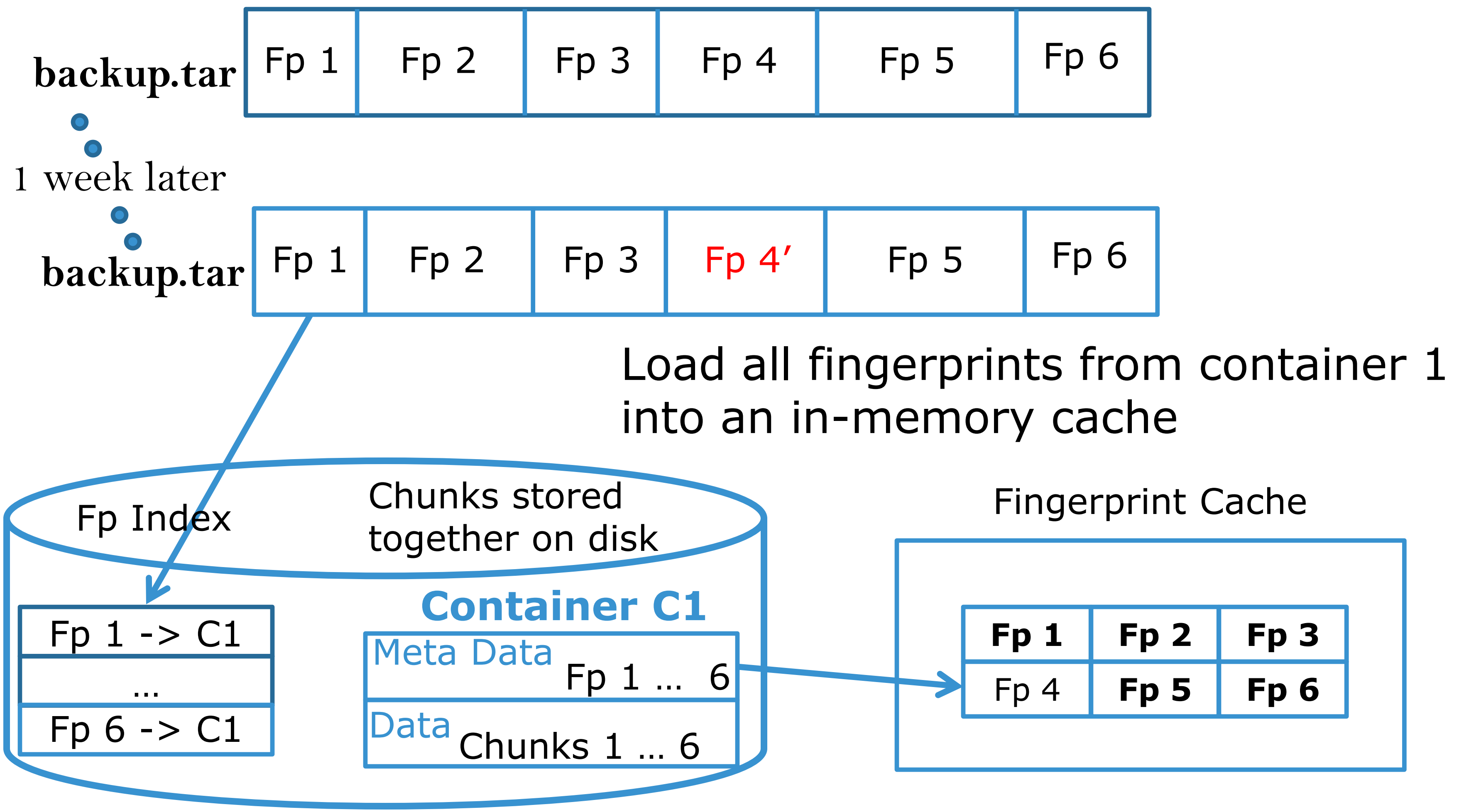


EMC²

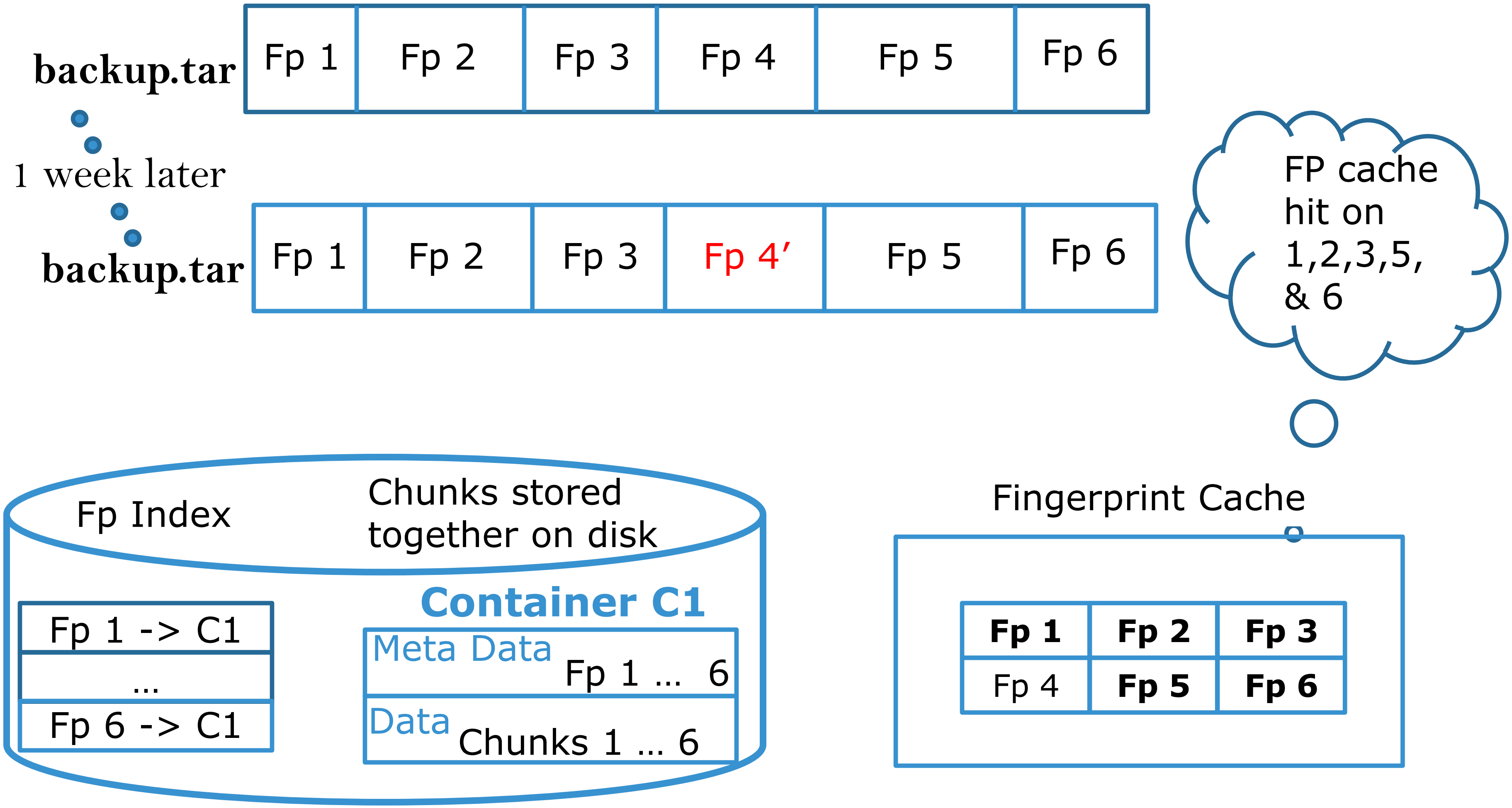
Stream-informed Deduplication



Stream-informed Deduplication



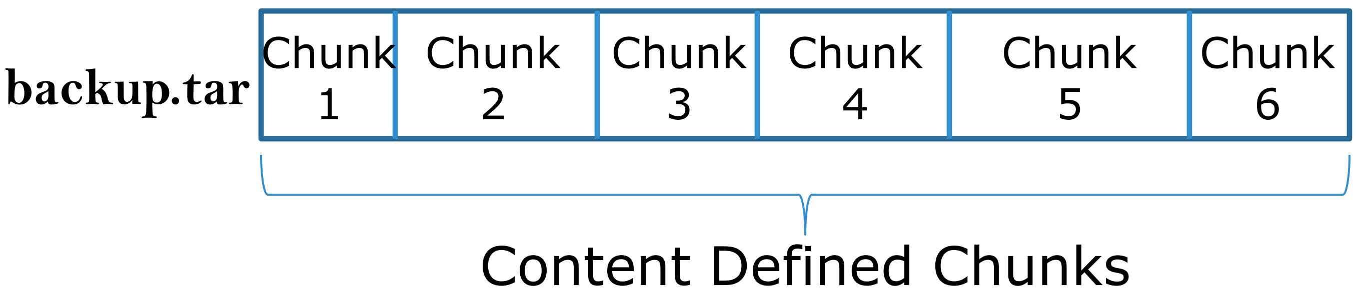
Stream-informed Deduplication



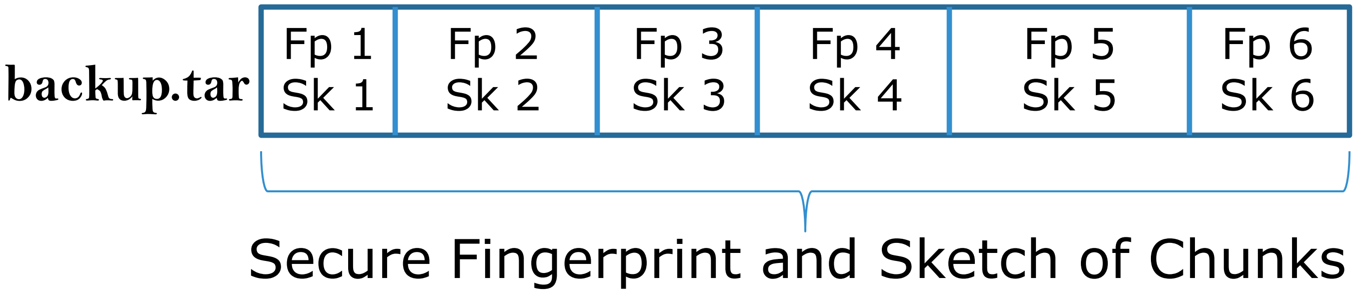
Stream-informed Deduplication and Delta Compression

backup.tar

Stream-informed Deduplication and Delta Compression

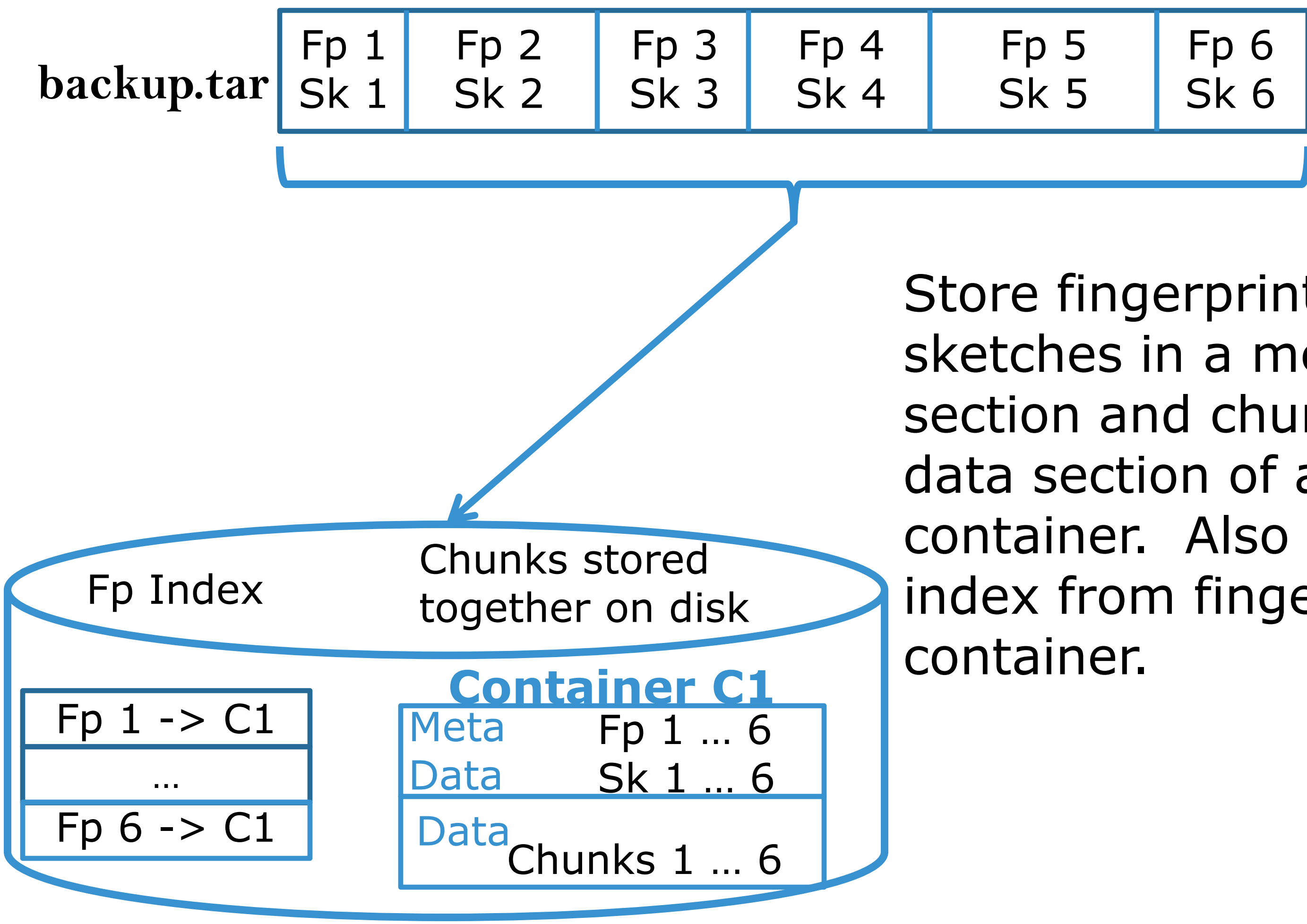


Stream-informed Deduplication and Delta Compression



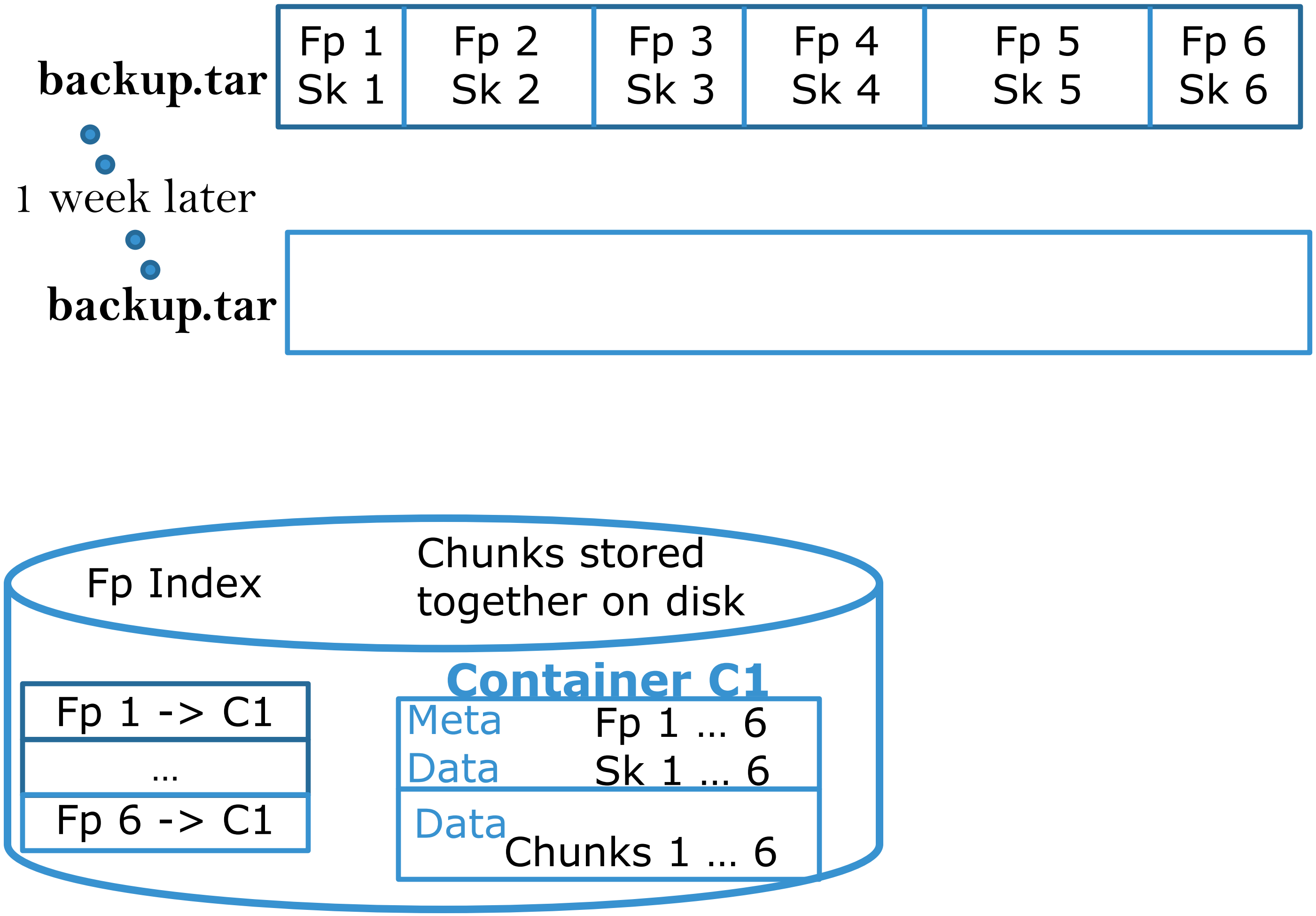
Calculate fingerprints used for deduplication and sketches used for similarity detection

Stream-informed Deduplication and Delta Compression

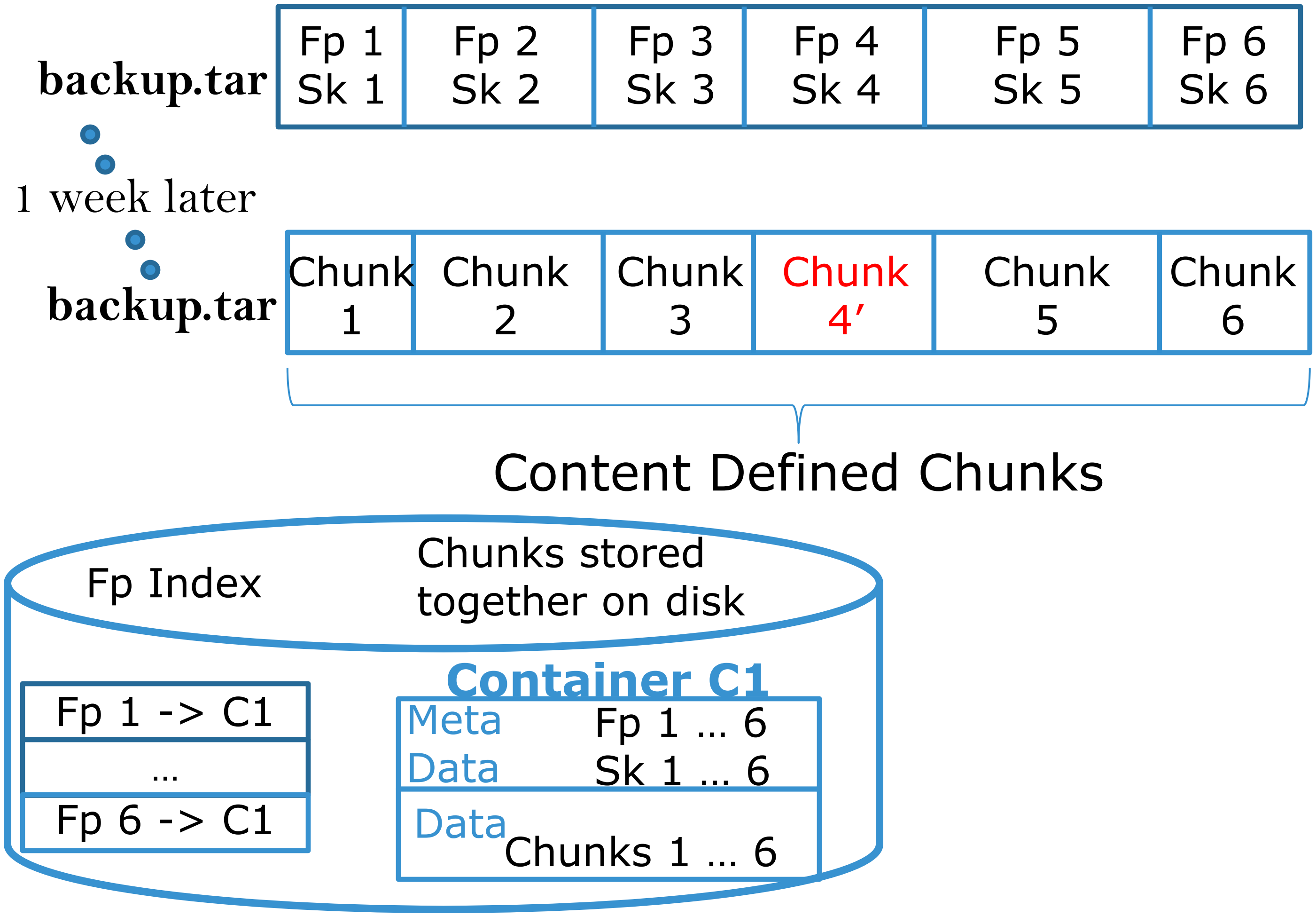


Store fingerprints and sketches in a metadata section and chunks to a data section of a container. Also create an index from fingerprint to container.

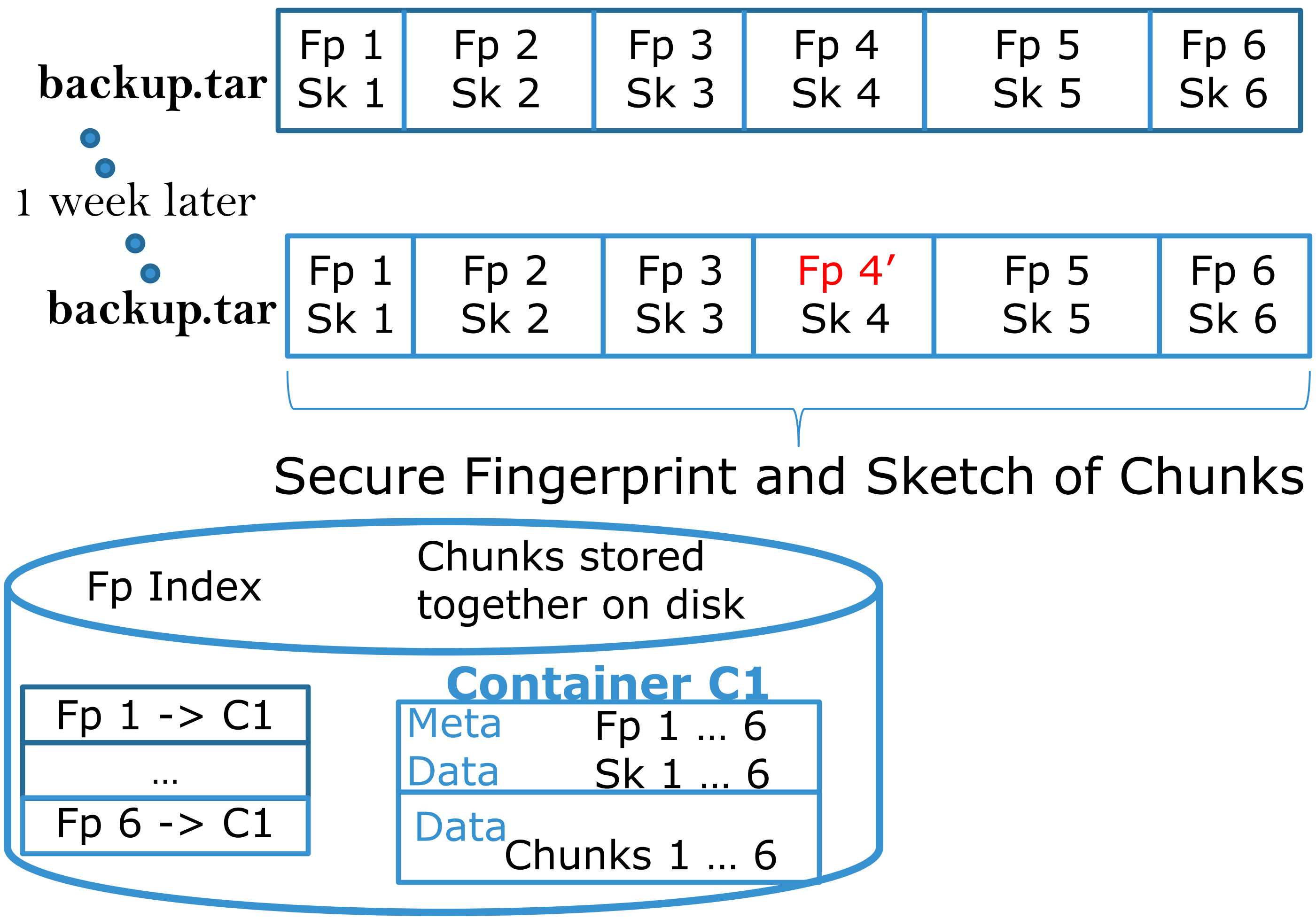
Stream-informed Deduplication and Delta Compression



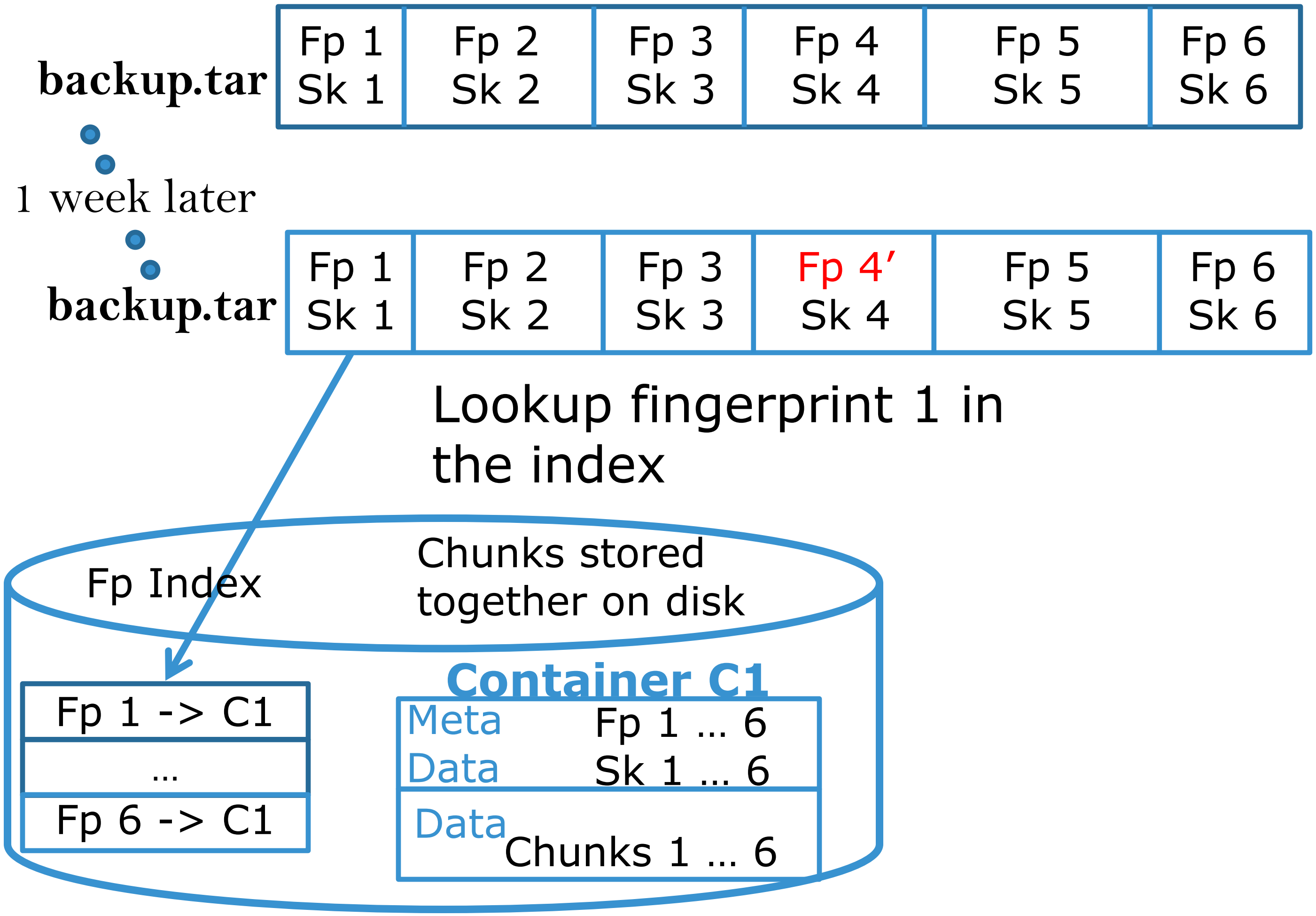
Stream-informed Deduplication and Delta Compression



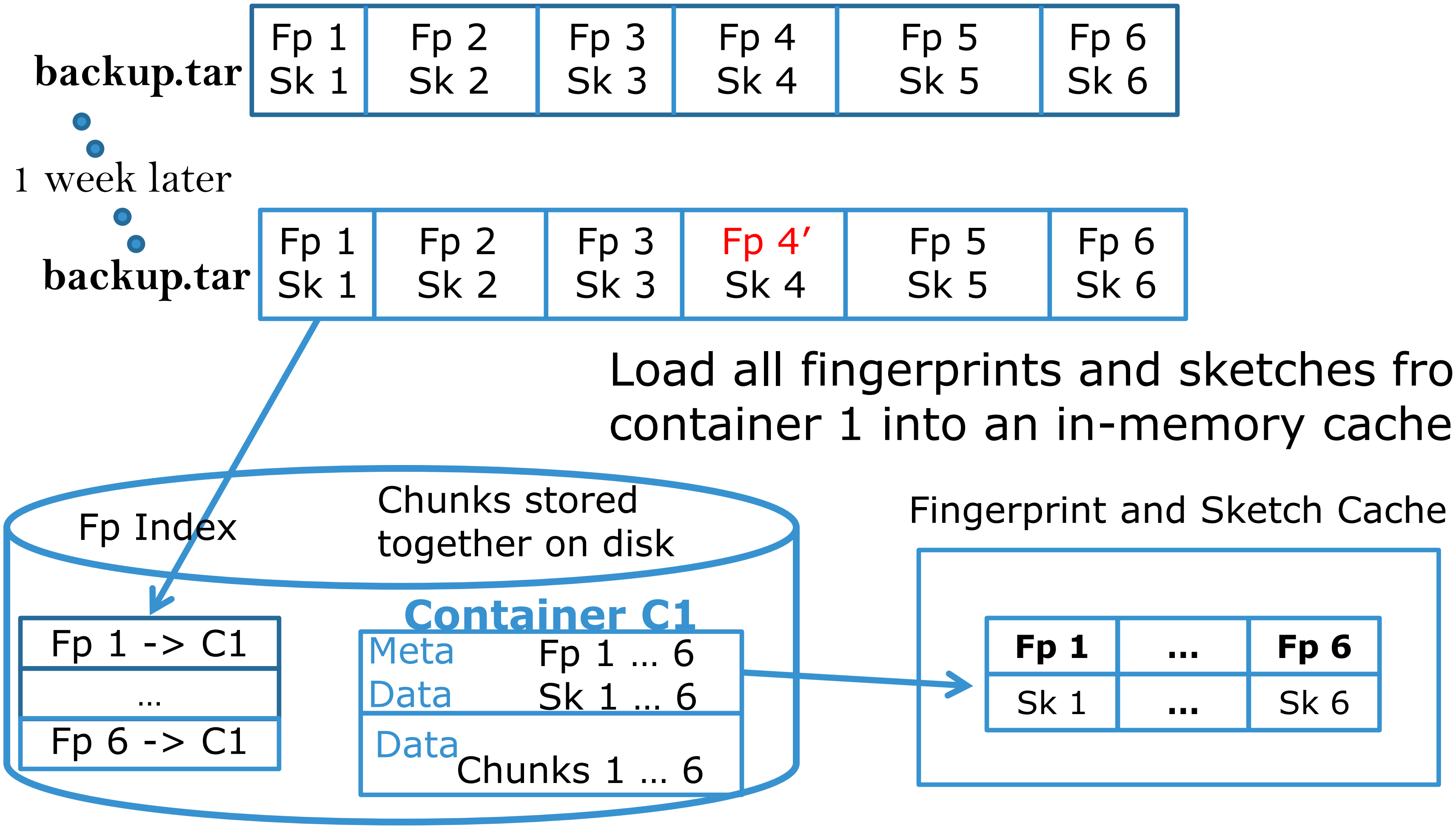
Stream-informed Deduplication and Delta Compression



Stream-informed Deduplication and Delta Compression

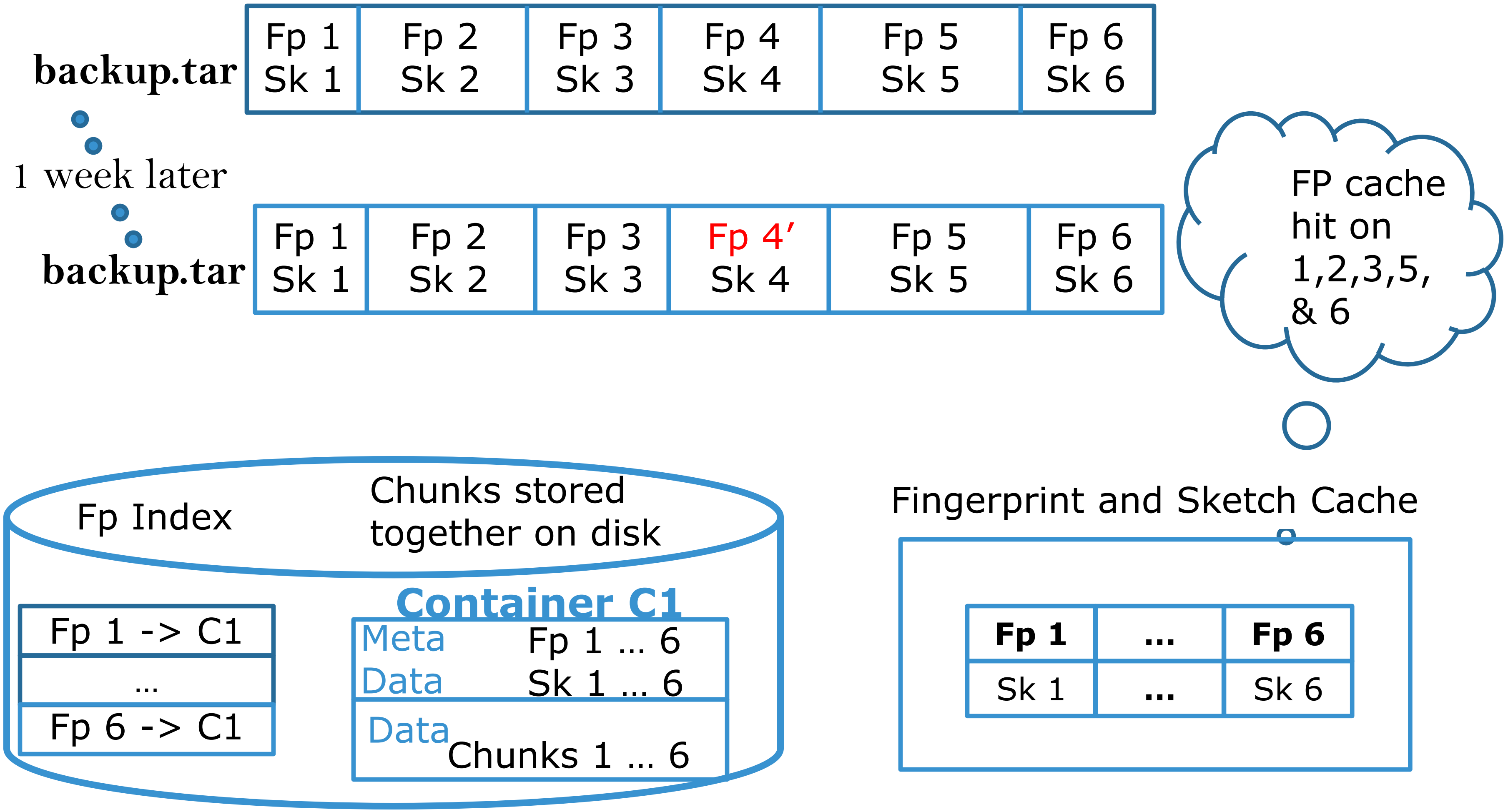


Stream-informed Deduplication and Delta Compression

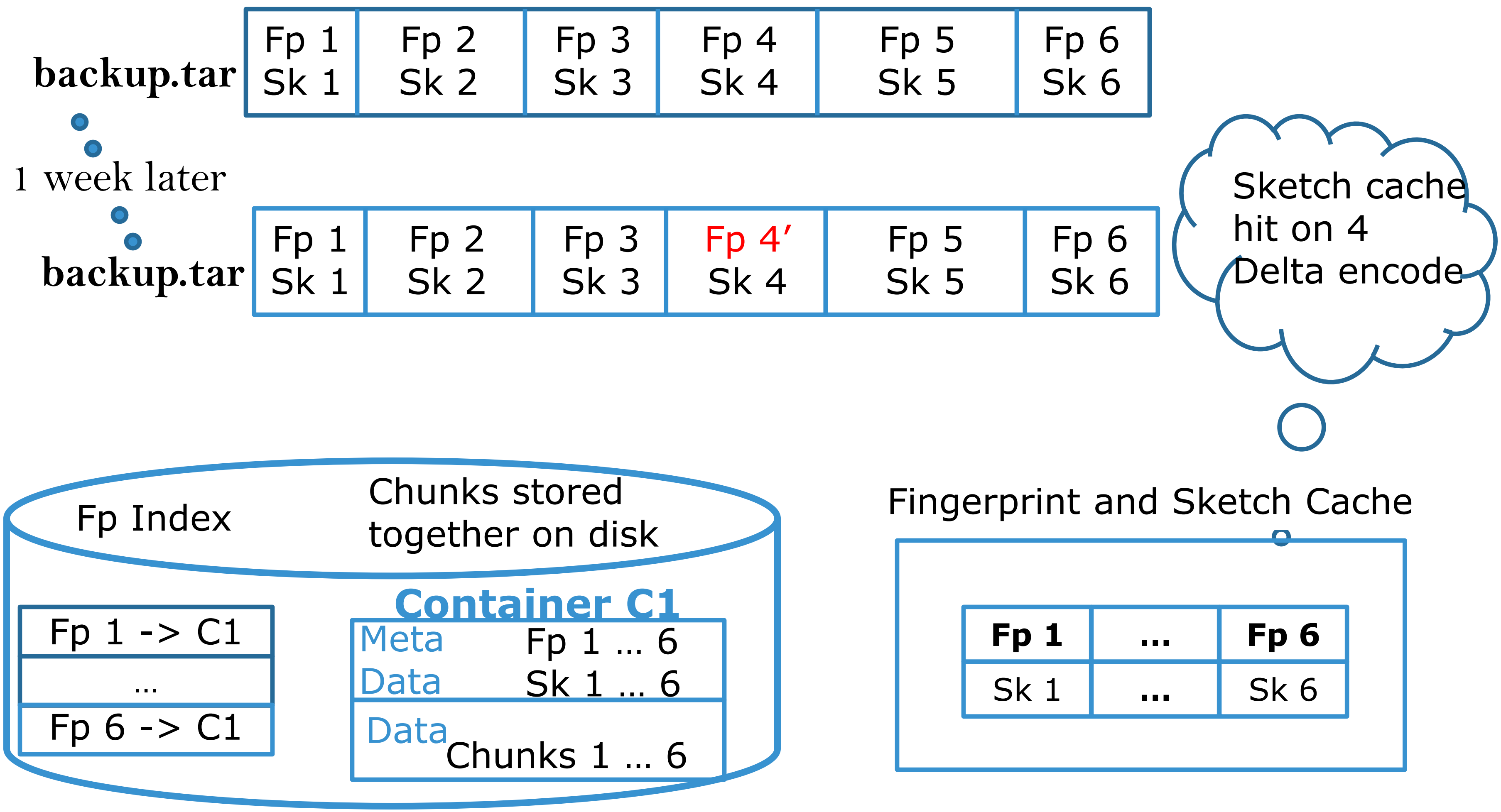


Load all fingerprints and sketches from container 1 into an in-memory cache

Stream-informed Deduplication and Delta Compression

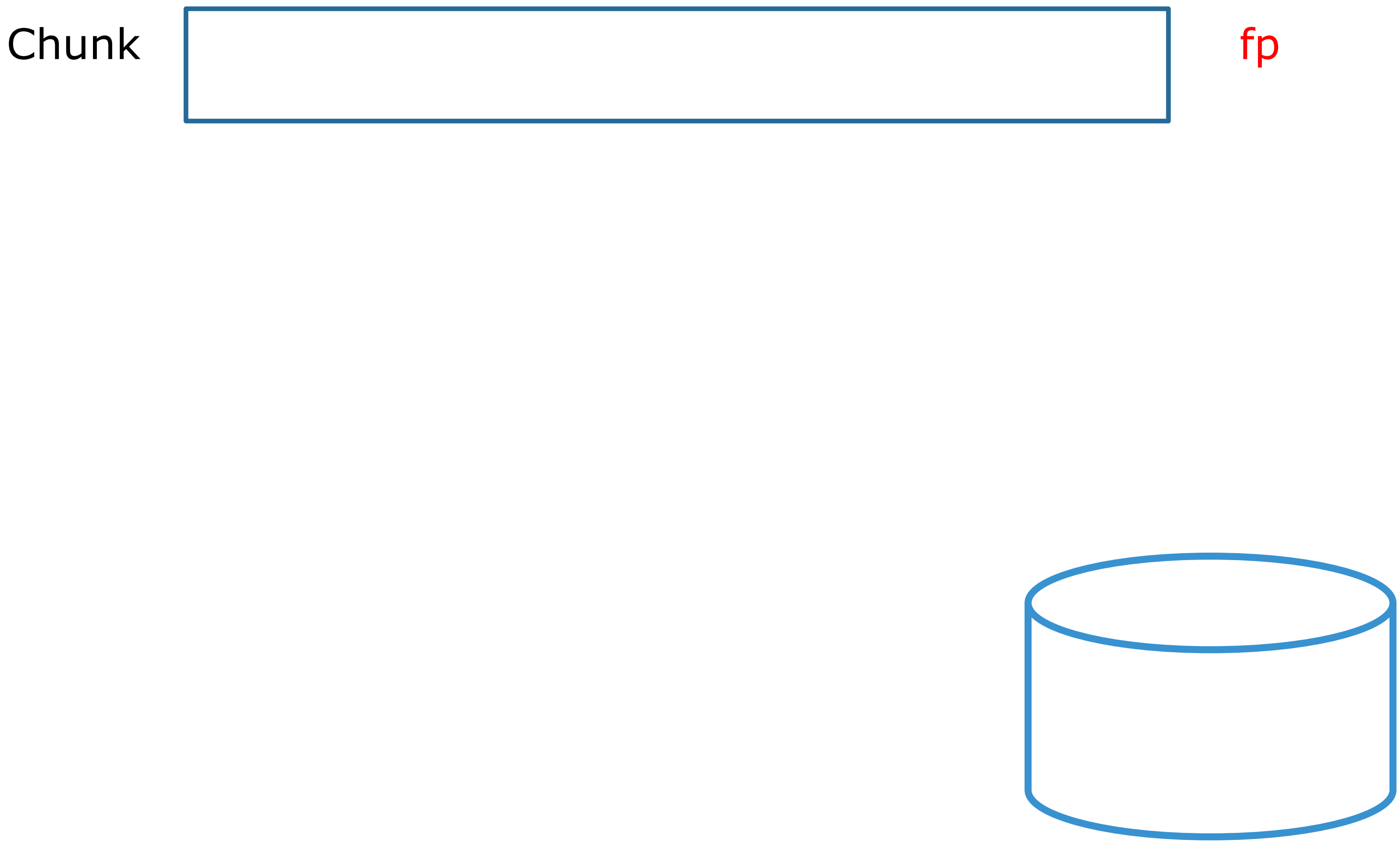


Stream-informed Deduplication and Delta Compression



EMC²

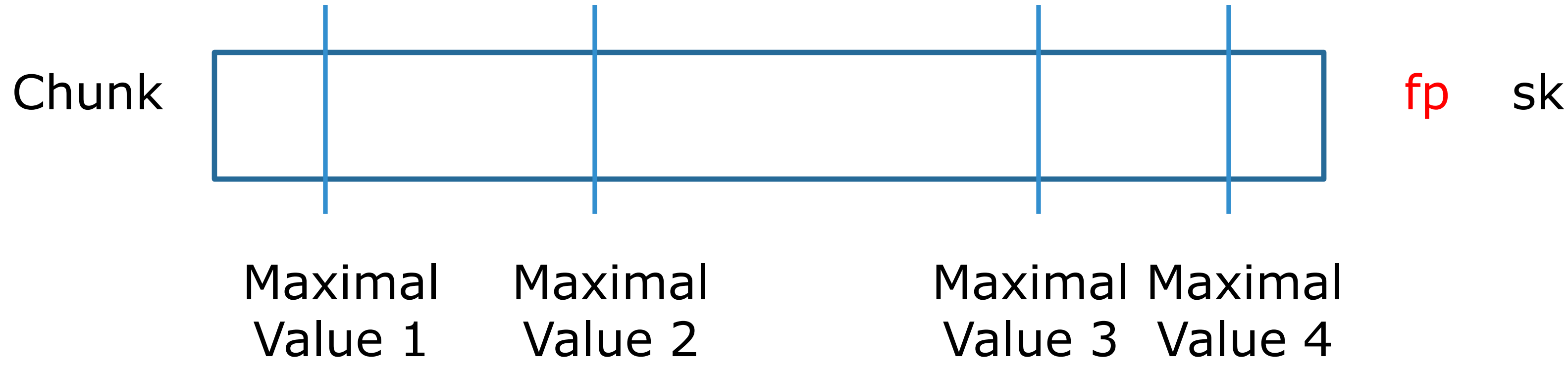
Deduplication and Delta Compression



Sketches based on Broder’97

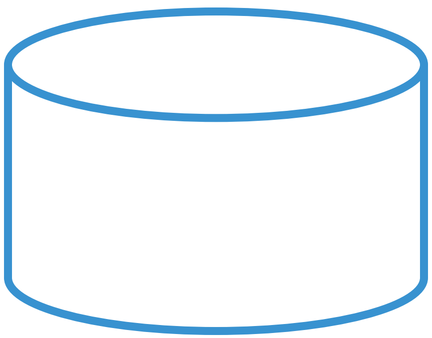
EMC²

Deduplication and Delta Compression



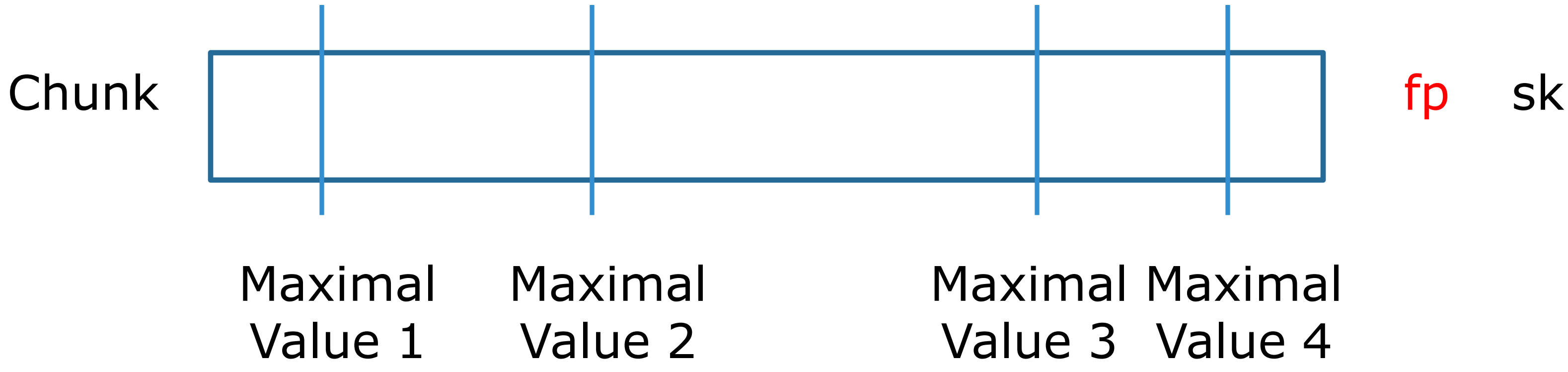
super_feature = Rabin_fp($feature_1$…$feature_4$)

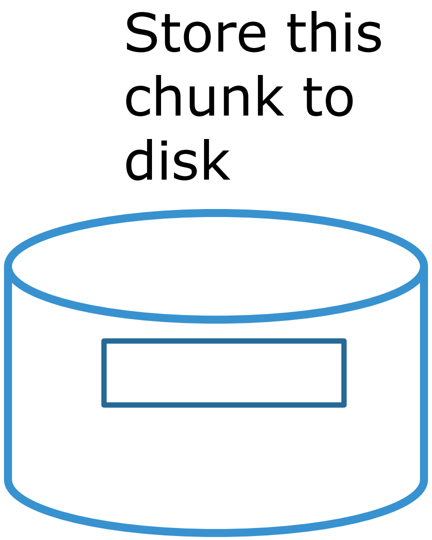
sketch is one or more super_features



Sketches based on Broder'97

Deduplication and Delta Compression





$$\text{super\_feature} = \text{Rabin\_fp}(\text{feature}_1 \ldots \text{feature}_4)$$

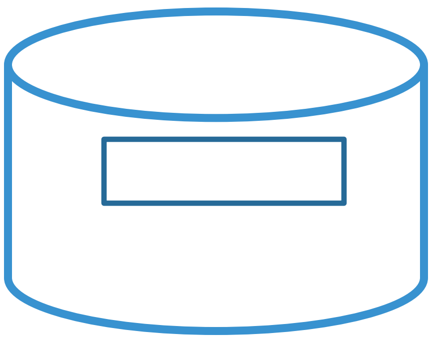
sketch is one or more super_features

Sketches based on Broder'97

Deduplication and Delta Compression

(duplicate of earlier chunk)

Fingerprint is a match, so do not store



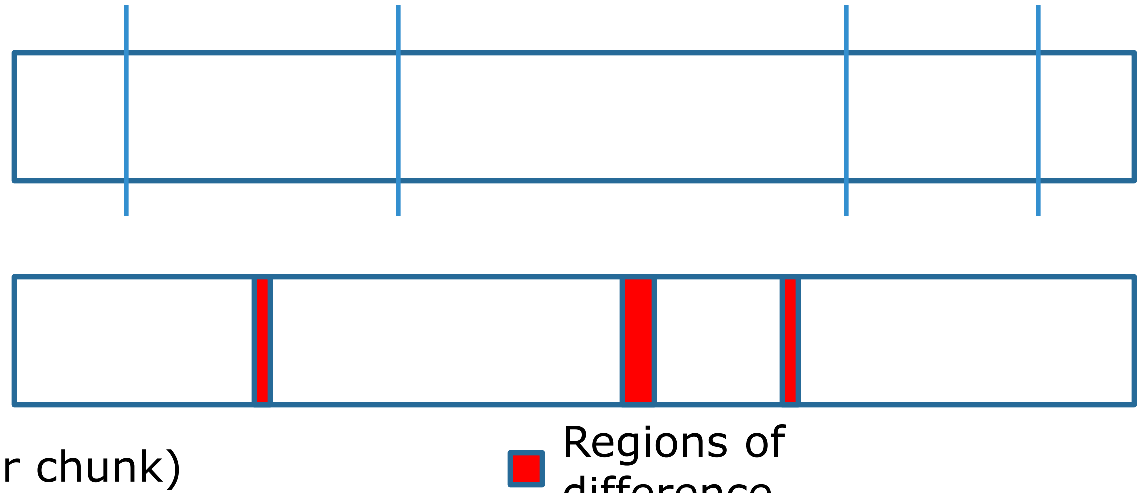
super_feature = Rabin_fp($\text{feature}_1 \ldots \text{feature}_4$)

sketch is one or more super_features

Sketches based on Broder'97

Deduplication and Delta Compression

Chunk



fp sk

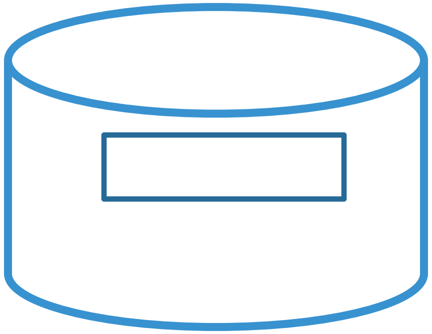
Chunk

(similar to earlier chunk)

Regions of difference

super_feature = Rabin_fp($feature_1$…$feature_4$)

sketch is one or more super_features

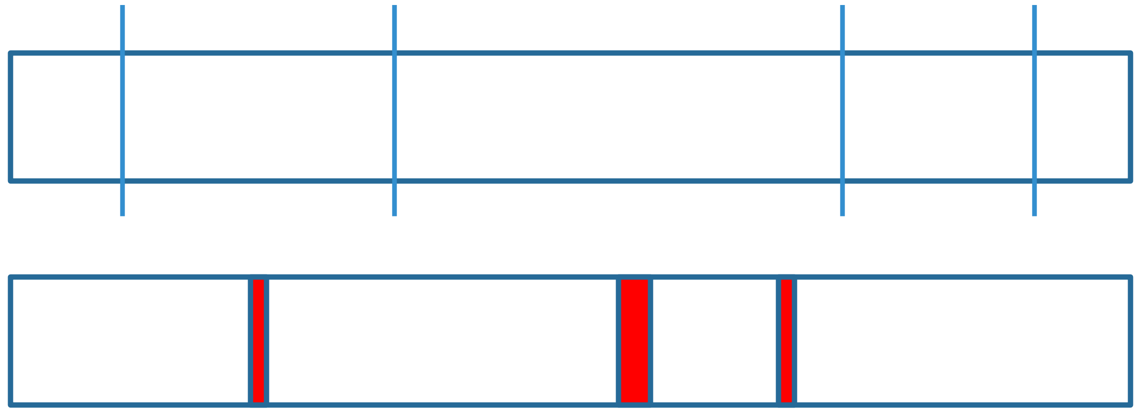


Sketches based on Broder'97

EMC²

Deduplication and Delta Compression

Chunk



fp sk

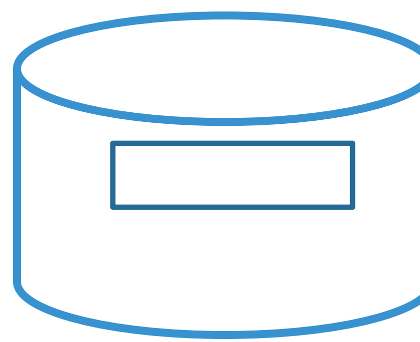
Chunk

fp

(similar to earlier chunk)

Regions of difference

Fingerprint is not a match, so calculate a sketch

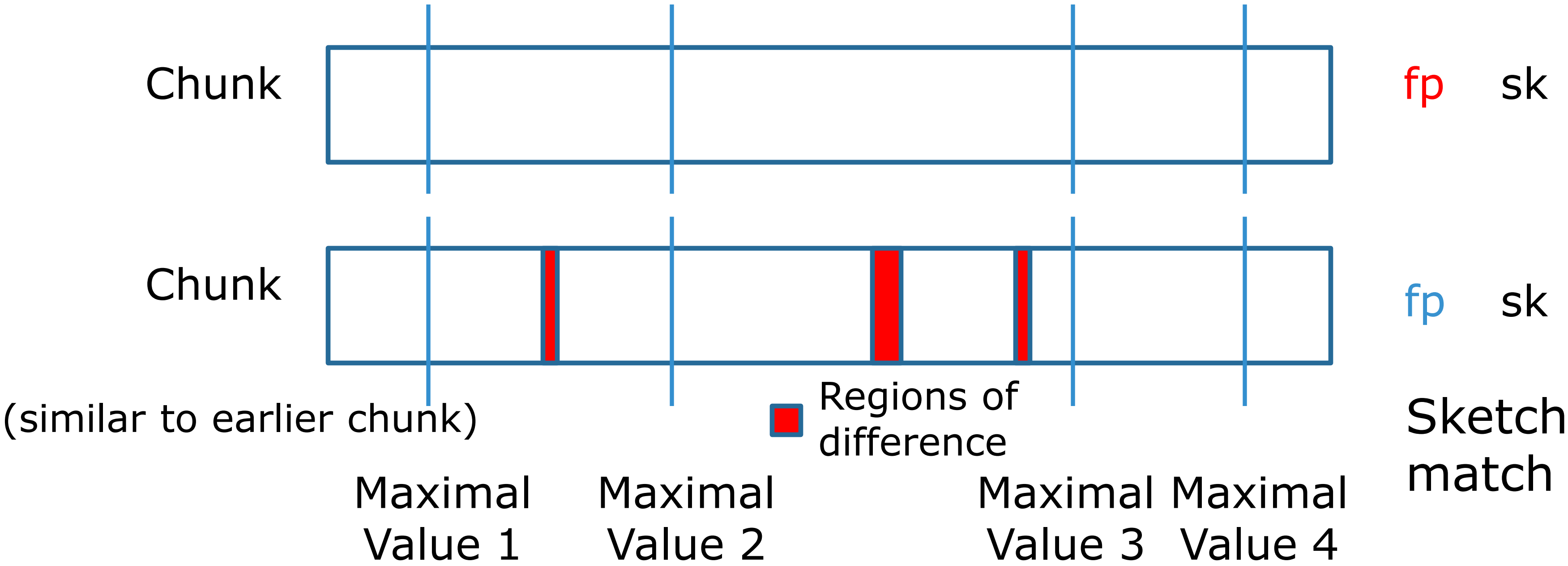


super_feature = Rabin_fp(feature$_1$…feature$_4$)

sketch is one or more super_features

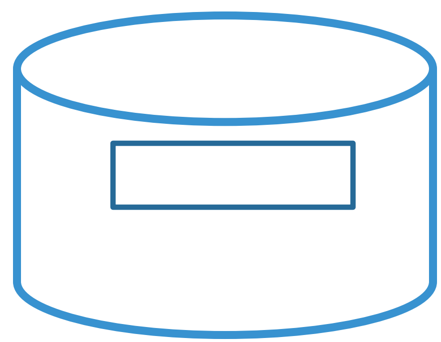
Sketches based on Broder'97

Deduplication and Delta Compression



super_feature = Rabin_fp(feature$_1$…feature$_4$)

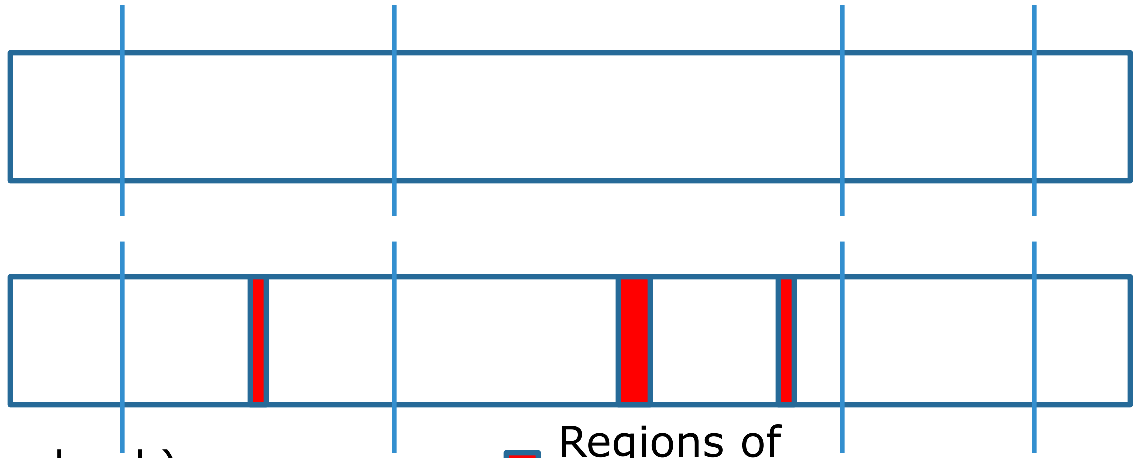
sketch is one or more super_features



Sketches based on Broder'97

Deduplication and Delta Compression

Chunk



fp sk

Chunk

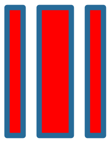
fp sk

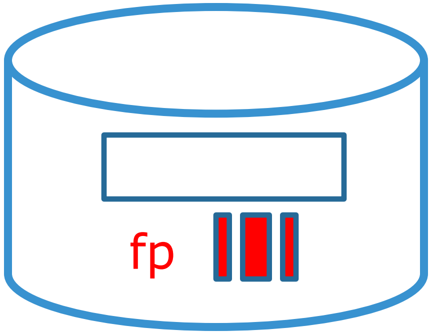
(similar to earlier chunk)

 Regions of difference

Calculate a delta and store the changed bytes and a reference to the earlier chunk

Store fp and differences





Sketches based on Broder'97

EMC²

Backup Datasets

Dataset	Type	Backup Policy	TB	Months
Workstations	16 desktops	Weekly full Daily incremental	2.3	4
Email	MS Exchange server	Daily full	2.5	5
Source Code	Version control repository	Weekly full Daily incremental	4.5	6
System Logs	Server's /var directory	Weekly full Daily incremental	5.3	4

EMC²

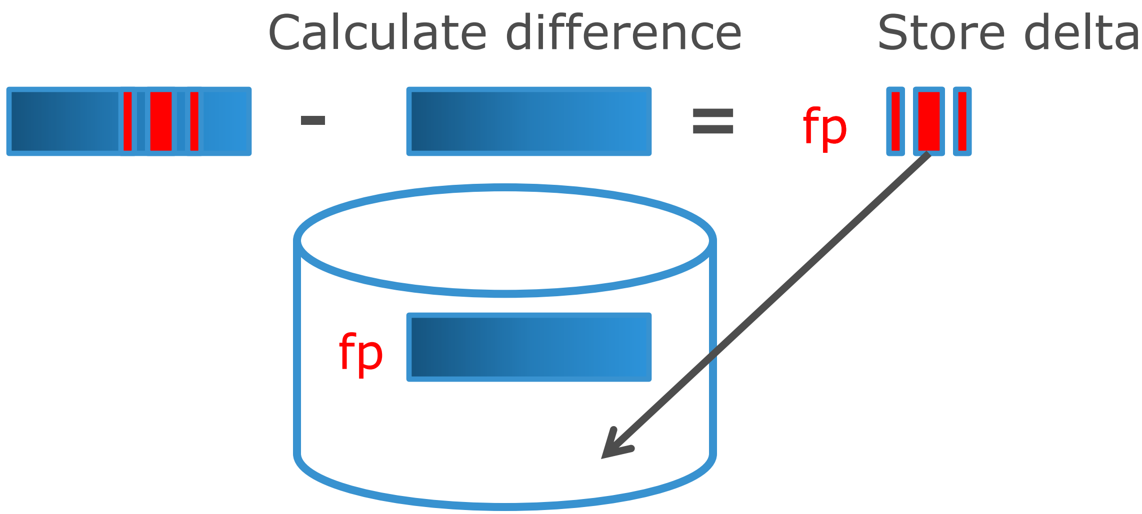
Compression Results

Delta adds 1.4 – 3.5X compression improvement over deduplication and LZ

Dataset	Deduplication	Delta	LZ	Total Compression	Delta Improvement
Workstations	5.0	4.2	1.6	33.6	3.5
Email	4.9	2.6	2.1	26.8	2.1
Source Code	16.7	3.6	2.5	150.3	1.4
System Logs	25.2	3.3	1.8	149.7	1.5

Throughput

- Delta compression requires extra computation and I/O
- Compare to deduplicated storage as baseline
- Throughput: 74% on first full backup
  Throughput: 53% on later full backups



Throughput Stages

I/O to a hard drive is the bottleneck, switching to SSD may help

Single-stream timing for each stage

Dataset	Sketch MB/s	Lookup MB/s	Encode Mb/s	Read Alternatives HDD MB/s	Read Alternatives SSD MB/s
Workstations	47	1,528	94	5	400
Email	49	1,441	69	1	80
Source Code	30	30	31	2	160
System Logs	30	70	50	2	160

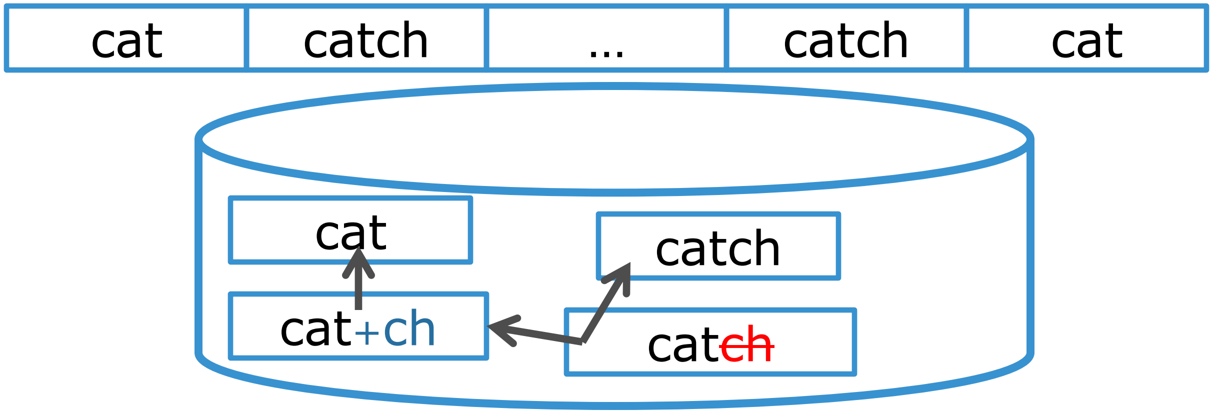
Aggregate throughput is higher due to: deduplication, 2/3 are delta encoded, multi-threading and asynchronous reads across multiple disks

EMC[2]

Indirection Complexities

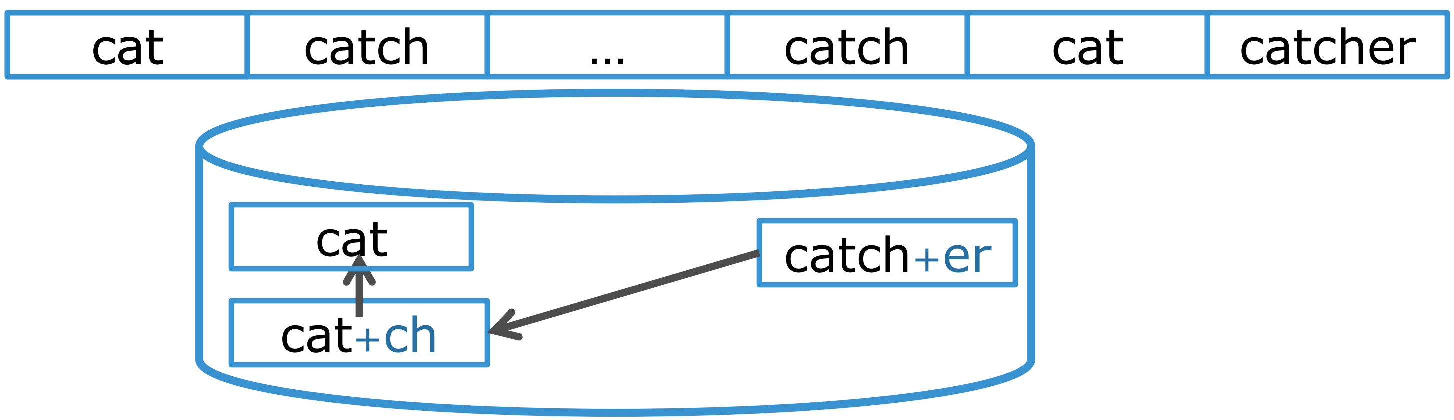
- Writing duplicates causes unintended read paths
  - Unpredictable read back times

Read back cat: There are multiple options, some that involve delta references



Indirection Complexities

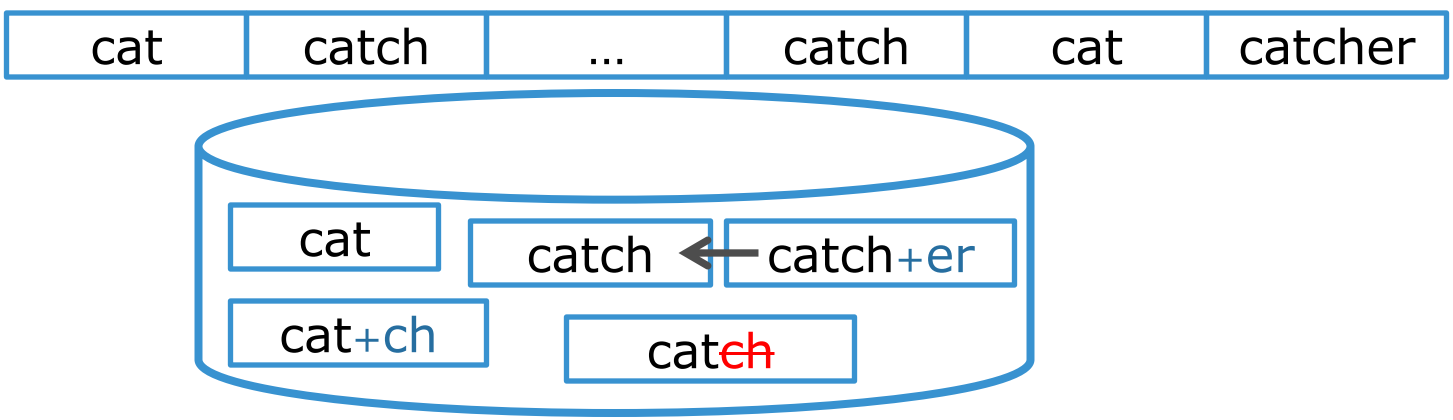
- Writing duplicates causes unintended read paths
  - Unpredictable read back times
- Multi-level delta increases compression and complexity
  - We implemented 1-level delta



Indirection Complexities

- End-to-end validity checks are slow because of remote references
  - Reconstructing a delta chunk requires reading the base

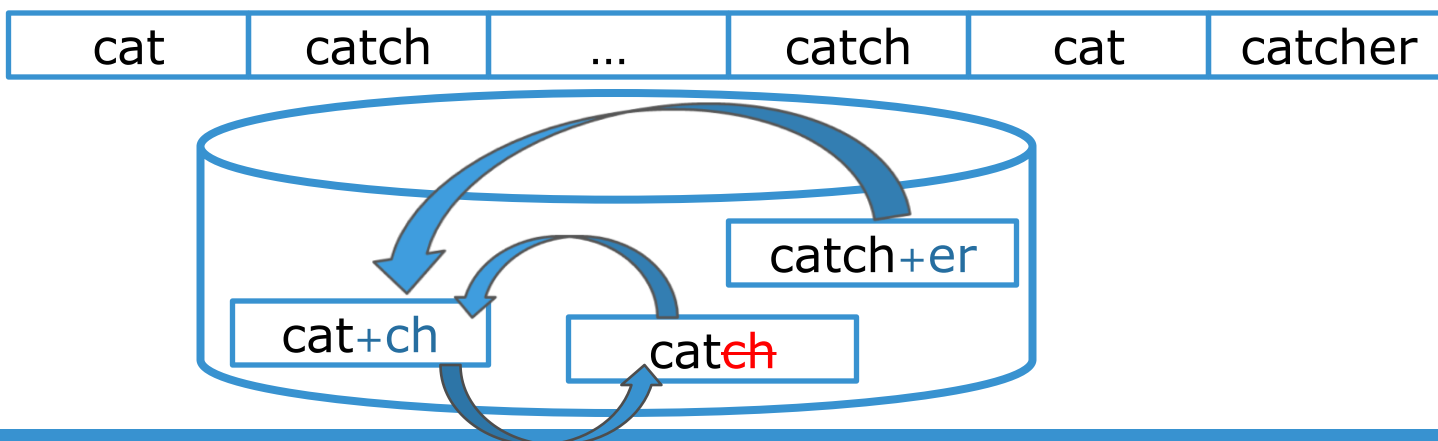
Verify catcher



Indirection Complexities

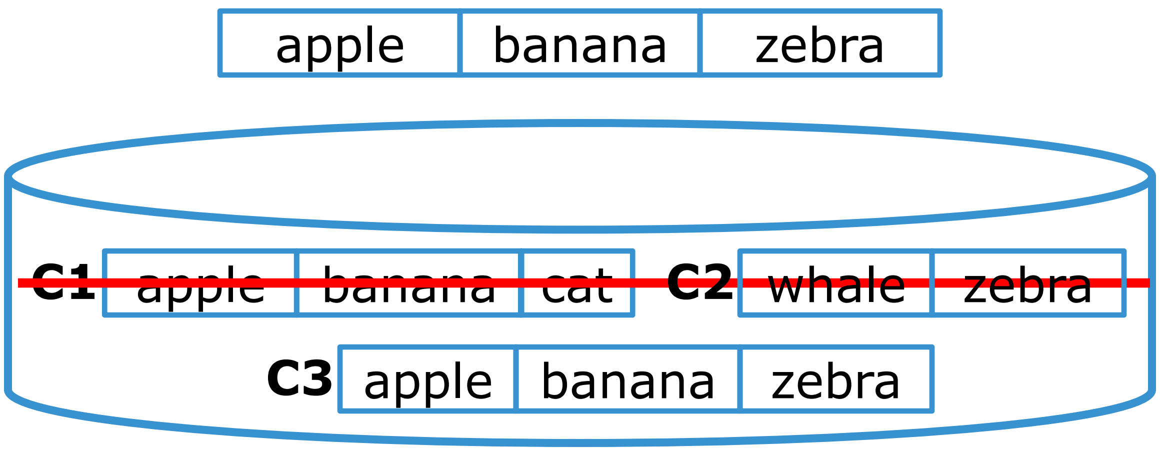
- End-to-end validity checks are slow because of remote references
  - Reconstructing a delta chunk requires reading the base
- Incorrect garbage collection can cause loops and data loss

Verify catcher: Loop due to incorrect GC indicates data loss



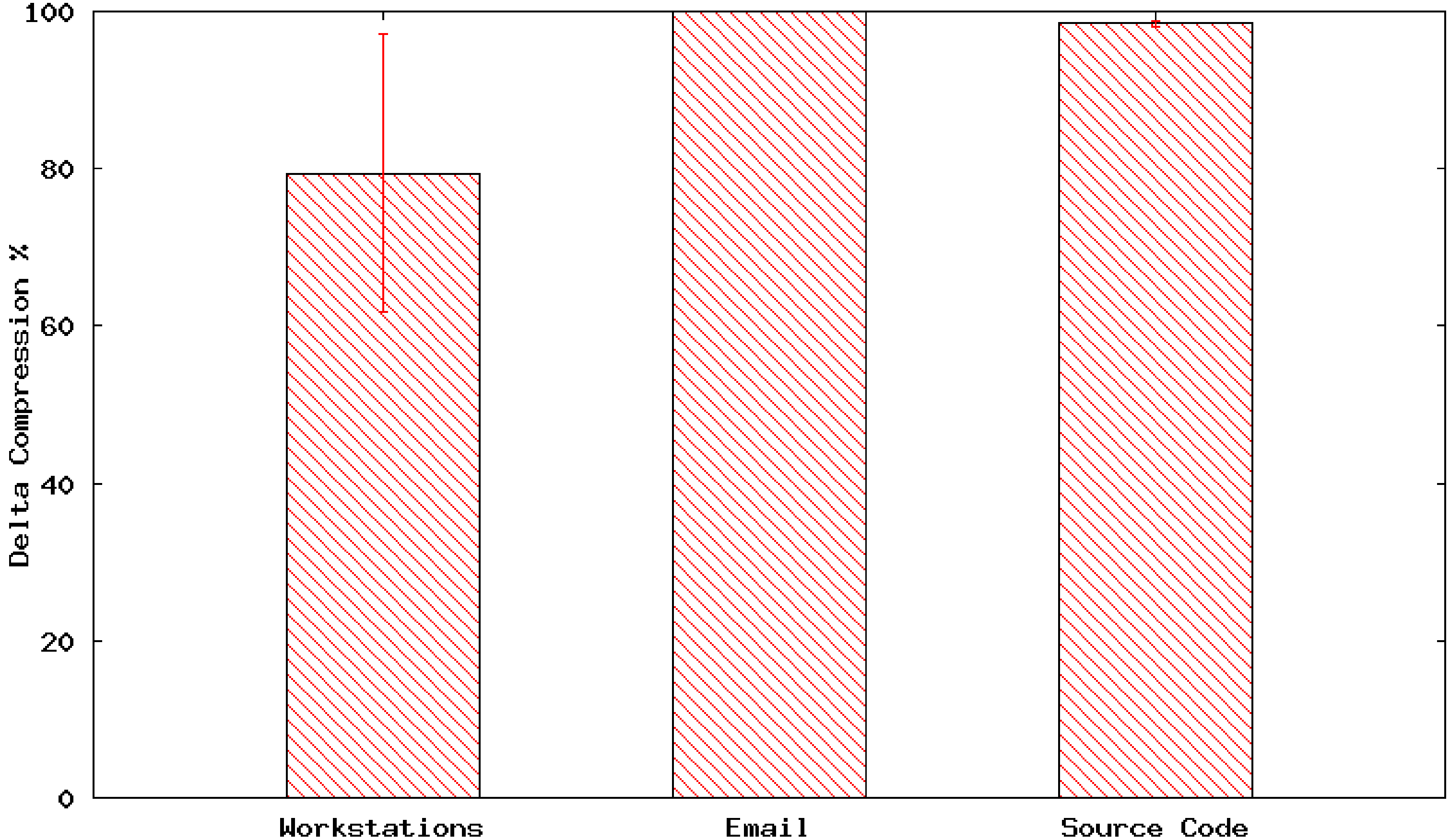
Garbage Collection

- Cleaning deleted chunks in a log structured file system
  - Reference counts
  - Mark-and-sweep
- Copying live chunks forward changes data locality, which impacts delta compression



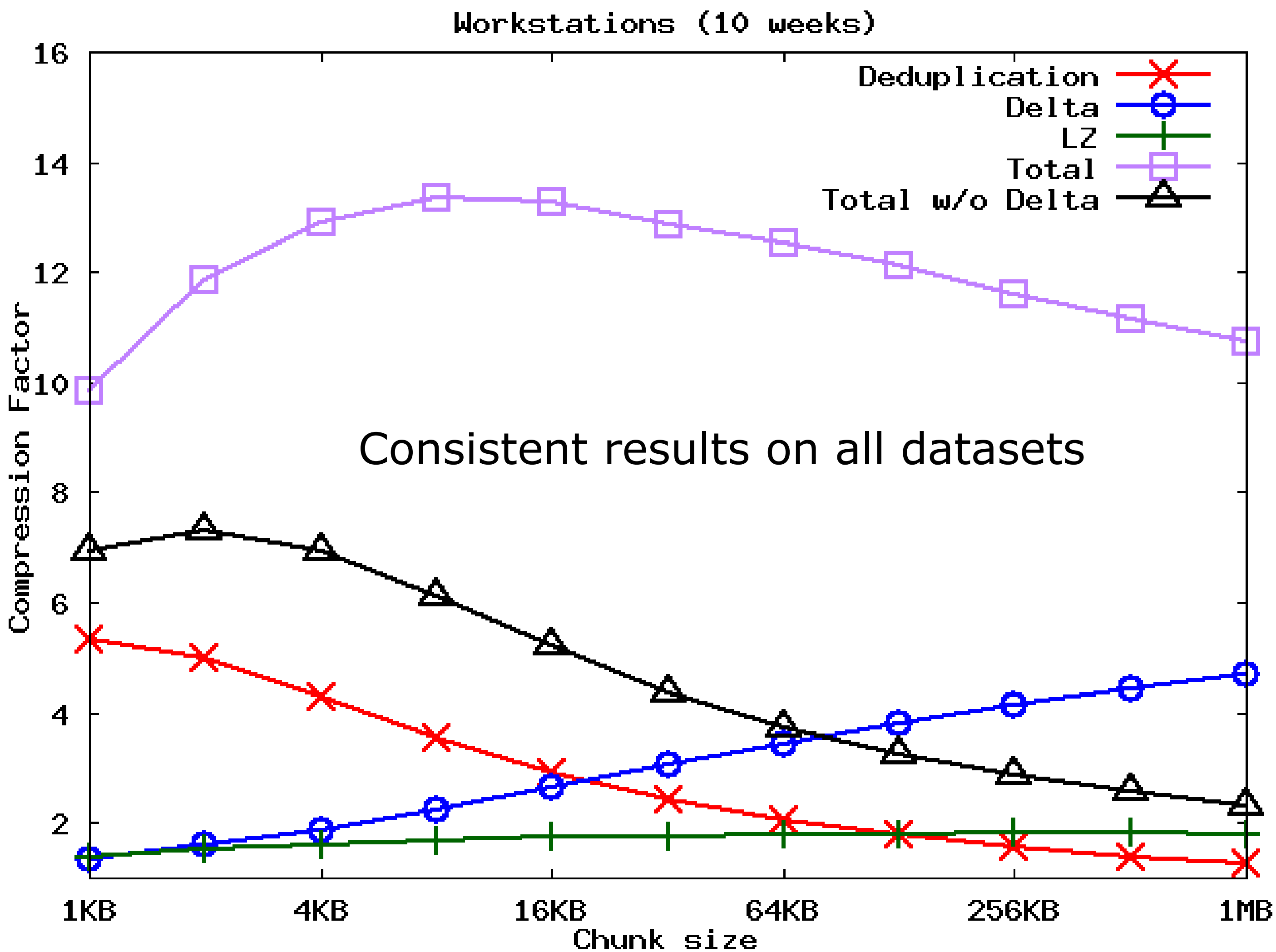
Garbage Collection Impact on Delta Compression

GC changes data locality but has only a small impact on delta compression



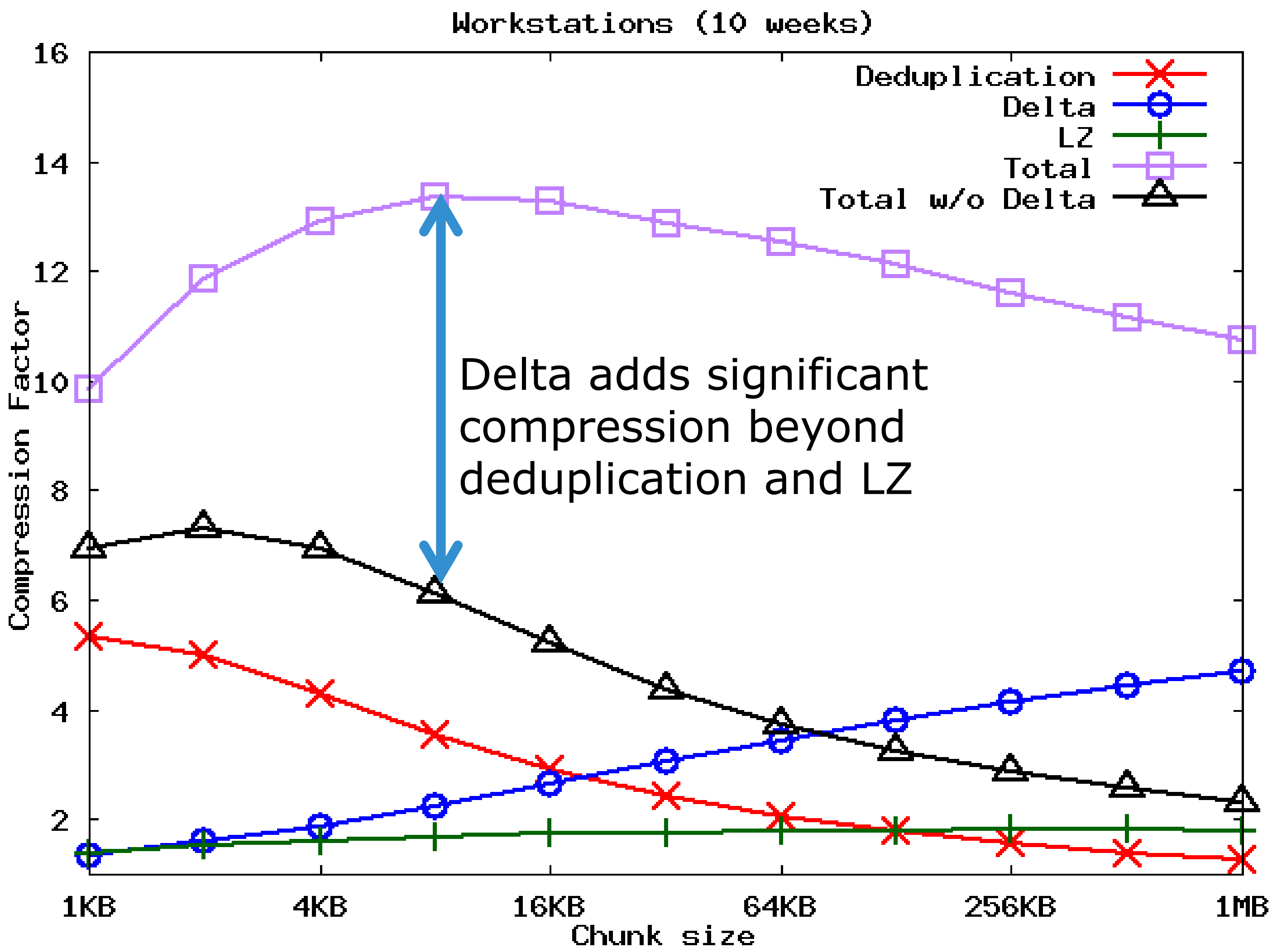
EMC²

Compression vs. Chunk Size

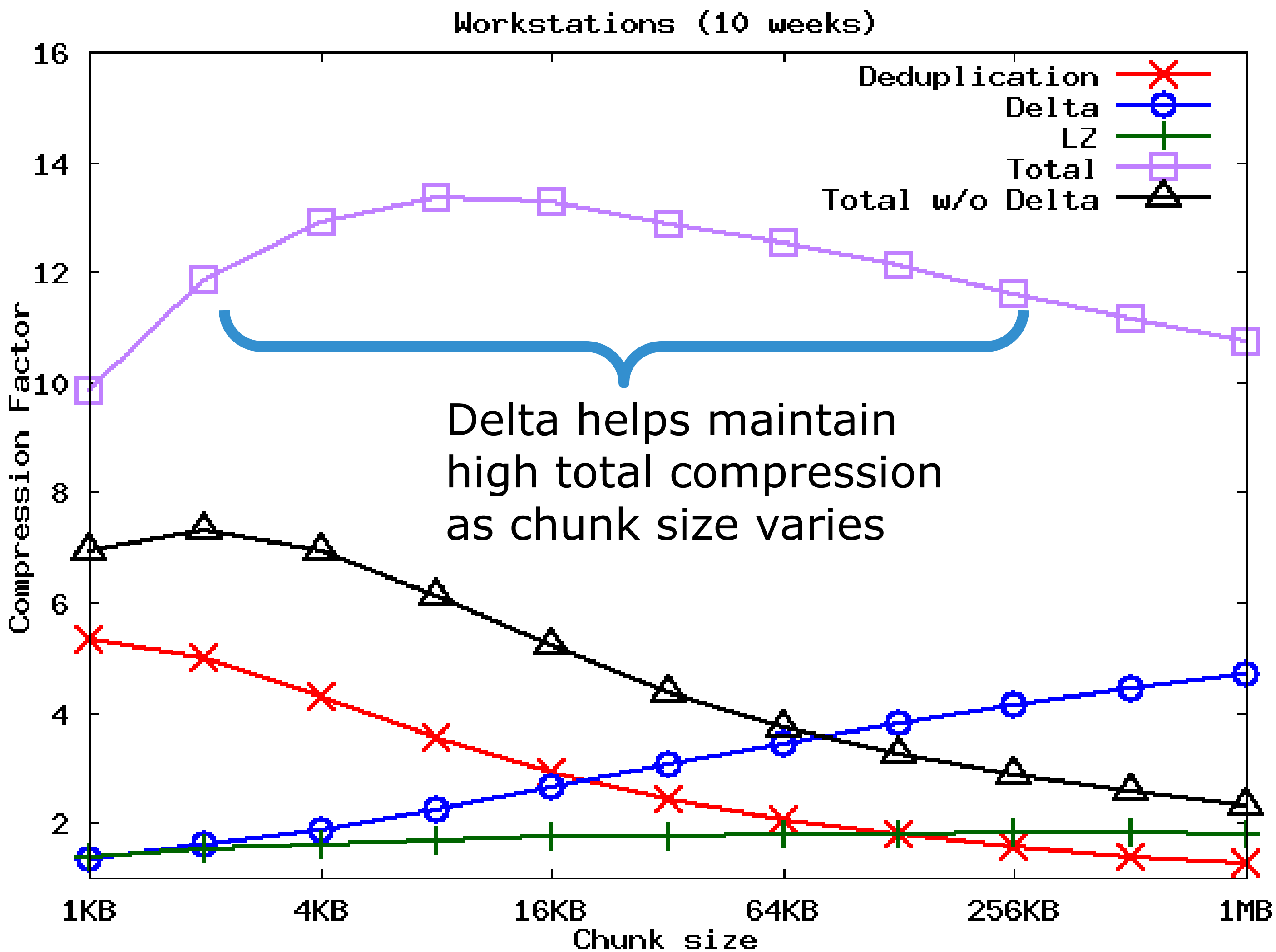


Consistent results on all datasets

Compression vs. Chunk Size



Compression vs. Chunk Size



Summary and Future Work

- Deduplication and delta compression prototype
  - Stream-informed locality replaces sketch indexes and improves write path throughput
  - Adds 1.4X – 3.5X compression
- Studied throughput
  - Throughput 50% of underlying deduplication system
  - Areas for improvement: SSD
- Garbage collection and data integrity
  - Remote reference complexity
  - Affects speed and validity
- Delta helps maintain overall compression across a broad range of chunk sizes

EMC²

Questions?

EMC²