TAGGED CONSISTENCY AND GARBAGE IDENTIFICATION IN DEDUPLICATION-ENABLED STORAGE SYSTEMS

Awais Khan, Chang-Gyu Lee, Sungyoung Park, Youngjae Kim
Sogang University, Seoul, Republic of Korea
Email: (awais, changgyu, parksy, youkim)@sogang.ac.kr

INTRODUCTION

- Increasing trend in adoption of data deduplication to improve disk space saving and reducing storage cost
- Deduplication introduces architectural transformations at transaction level, however disk failures and network crashes can lead system to inconsistent state
- Failure can cause invalid or mis-leading reference in deduplication metadata (hash-table).
- Duplicate entries can refer (increment, decrement) an invalid or mis-leading entry in hash-table.
- Garbage (data chunks) left of the failed transactions cannot be collected efficiently and result in poor disk space saving.

MOTIVATION & RESEARCH CHALLENGES

- Deduplication metadata if not properly handled can threaten system consistency and data integrity.
- Failure are not common in deduplication, however, if not properly handled can cause two possible problems.
- First, Mis-leading Reference entry and false reference count value computed due to mis-leading entry.
- Second, Garbage Identification, how to efficiently identify which chunks are garbage (left of the failed transactions).
- Journaling and Check pointing can incur high overhead along with additional space requirements.
- High Performance Overhead and transaction blocking can occur.

PROPOSED SYSTEM ARCHITECTURE

- We propose tagged consistency mechanism to address the problems caused by failures in data deduplication-enabled storage systems.
- We present two variants of tagged consistency mechanism, each based on tag management at different level (chunks and objects).
- **Chunk-based Consistency:** In this approach, consistency flag is stored in Chunk info schema as shown in Figure 1 (a).
- By default, each chunk carry invalid flag in hash-table. Once chunk is stored successfully, an additional I/O is sent to deduplication service to switch the flag.
- When duplicate chunk is detected, before changing the reference count (increment or decrement), we check the flag of chunk. If the flag is valid, then change is permitted else we undergo a Reconciliation Check, which checks the status of chunk is storage system.
- If chunk is present in storage system, we switch the flag and carry-out reference count operation (increment/decrement operation) if the chunk is not stored, we store the data chunk and then switch the flag.
- **Object-based Consistency:** This approach stores flag in Object Map schema as shown in Figure 1 (a) and requires only single additional I/O at transaction completion to switch the consistency tags.
- Unlike previous one, this approach includes only one additional I/O per I/O transaction (Object-level). The rest of transaction follows the same pattern, increment and decrement will require flag check before.
- This approach is expected to have better performance because of only a single additional I/O required to switch flag.

TAGGED CONSISTENCY OVERVIEW

![Figure 1. Proposed Tagged Consistency Overview.](image)

GARBAGE IDENTIFICATION

- The garbage identification and removal is a challenging problem in existing data deduplication frameworks.
- The existing approaches such Journaling, Group Mark and Sweep have shown different monitoring and logging mechanisms to collect garbages.
- However, these approaches suits well to backup storages and not primary storage system because of idle time and resource availability requirements.
- The proposed garbage identification mechanism does not require any additional and complex implementation, rather single query can provide list of garbages.
- We collect all the chunks with inconsistent flag and wait up to certain configurable time threshold, if no change occurs to collected entries, we mark these chunks as garbage chunks and remove from storage system.

PRELIMINARY EVALUATION

- We used Fio benchmark tool to evaluate the proposed tagged consistency mechanism. We used Ceph Cluster consisting of one monitor and two OSD servers, each equipped with two SSDs.
- Preliminary evaluation shows that tagged consistency can incur overhead even in case when chunk size is big.
- However, Object-based consistency performed better than chunk because of single additional I/O whereas, chunk-consistency incurs very high overhead when chunk size is small due to additional I/O per chunk.
- Smaller chunk size can increase the chances of high disk space saving as compared to big chunk size.

CONCLUSION & FUTURE WORK

- Tagged Consistency ensures the system reliability and data integrity at certain overhead cost.
- Our future work includes the implementation of async communication between storage component (disk or OSD) and deduplication engine with zero performance overhead as compared to baseline.