**IBD distribution Read Cache**

**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **DATE** | **AUTHOR** | **REVISION** | **HISTORY LOG** |
| 1/21/16  1/29/16  2/4/16 | Guanheng Liu  Guanheng Liu  Guanheng Liu | A | Created  Updated more detail graphics  Strategy & Scalability |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

[1 Overview 3](#_Toc442367183)

[1.1 Requirement 3](#_Toc442367184)

[1.2 Scopes: 3](#_Toc442367185)

[1.3 Read pipeline 4](#_Toc442367186)

[1.3.1 CRC 4](#_Toc442367187)

[1.3.2 DRC 6](#_Toc442367188)

[2 Design 7](#_Toc442367189)

[2.1 Structure 7](#_Toc442367190)

[2.2 Strategy 8](#_Toc442367191)

[2.2.1 Balance 8](#_Toc442367192)

[2.2.2 Monotonicity 8](#_Toc442367193)

[2.2.3 Spread 8](#_Toc442367194)

[Mapping 8](#_Toc442367195)

[2.2.4 nse mapping (namespace to fp) 8](#_Toc442367196)

[2.2.5 dse mapping (fp to global drc) 8](#_Toc442367197)

[2.2.6 cse mapping (fp to data) 9](#_Toc442367198)

[2.3 Scalability 9](#_Toc442367199)

[2.3.1.1 Add Node 9](#_Toc442367200)

[2.3.2 9](#_Toc442367201)

[2.4 Synchronization 9](#_Toc442367202)

# Overview

This document covers design of distribution read cache (DRC) of IBD.

## Requirement

* Availability: Ibdserver can response read request with data locate at remote read cache node.
* Scalability: DRC node can be added or removed any time, data should able to migrate to other survival DRC
* Synchronization: DRC node should update the mapping table in time to provide distributed synchronization

## Scopes:

DRC Cache will locate at every ibdserver node as well as normal CAS read cache(CRC), ibdserver will handle how and where to access the correct DRC.

Read Request

ibdserver

ibdserver

ibdserver

ibdserver

MDS

Remote

Local

Read Cache

Globe Cache Mapping

ibdserver

Read Cache

Read Cache

Read Cache

……….

## Read pipeline

### CRC

Read Response

(data)

ibdagent

Read Request

(offset, length)

ibdserver

Try to read from write cache

If hit

wmc.c

(write cache)

If not hit

nse.c

(namespace to FP)

If not hit

If hit

cse.c

(FP to data)

Read Cache Device

MDS

### DRC

Read Response

(data)

ibdagent

Read Request

(offset, length)

ibdserver

Try to read from write cache

If hit

wmc.c

(write cache)

If not hit

If not hit

nse.c

(namespace to FP)

Remote ibdserver

cse.c

(FP to data)

If hit

dse.c

(globe DRC mapping table)

Read Cache Device

MDS

# Design

## Structure

For CRC, after searching finger printer mapping in nse, ibdserver will searching local namespace mapping in cse.

For DRC, there is a mid-layer between nse and cse called dse. Dse will contain and management globe mapping from finger print to globe read cache id. The globe mapping will store as a hash table \*. After a read request hit the nse mapping, it will search match finger print mapping in dse. DX/CX \* module will send the read request with finger print to remote ibdserver. Whenever the finger print hit the remote cse mapping, remote ibdserver will send back the read response with data which read from remote read cache to the local ibdserver.

## Strategy

### Balance

The globe hash algorithm should map the hash key(finger print) to hash value(Node ID) base on the size of Read Cache Node. As a result, the load will be as balance as possible.

e.g. Read Cache Node A is 10G, Read Cache Node B is 5G, Read Cache Node C is 5G, the hash map table should like:

0

Node C

Node B

Node A

Max Key

### Monotonicity

The hash key range will not remap to the old Node B when adding new Cache Node. In other word, there will no data transfer between old Nodes. The data transfer is only allowed from old Node to new Node.

### Spread

Some of the DRC Node may not be updated with the latest globe hash map version since network delay or brain split. The situation should be avoided as much as possible. Even it happened, the DRC will handle the issue and do not rise data corruption.

## Mapping

### nse mapping (namespace to fp)

nse.c will hold local mapping table from volume namespace from finger print. The mapping table only store in memory, the table will update after mwc flushing operation or two face read operation(when use mds as backing rescource).

### dse mapping (fp to global drc)

dse.c will hold global hash\* mapping table from finger print to drc node ip address. The mapping table need to back up on then hard disk for recover. The table will update when found a newer mapping table on remote node.

### cse mapping (fp to data)

cse.c will hold local hash mapping table from finger print to the namespace of read cache device. The mapping table need to back up on then hard disk for recover. The table will update only after mwc flushing operation.

## Scalability

AMC always have the latest globe DRC Nodes structure and globe hash map. However, after AMC generate the scale plan, DRC is able to handle scale-out & scale-in without AMC.

#### Add Node

1. All the new nodes receive the latest globe DRC nodes structure from AMC before joining the DRC group.

2. One node will be voted as Leader among all the new nodes, the leader node (LN) will calculate the new globe hash map base on the globe node structure.

3. The leader node broadcast the new hash map to all the DRC nodes, DRC node will switch to “to-LN” status after receive the message. The leader node will send the new hash map back to AMC if half of the node response the update request\*.

4. Execute the update plan including the hash mapping modification and the data transfer. Hash key will be the basic unit during the update. The update will be processed unit by unit. Whenever a hash key has been successfully re-mapped, switch its statue from “to-LN” to “LN”. The whole update process will not block the read request.

5 The leader node will notice the AMC after the Add Node Process finished.

### 

## Synchronization