COMMUNITY
THE ASF CONFERENCE
CODE

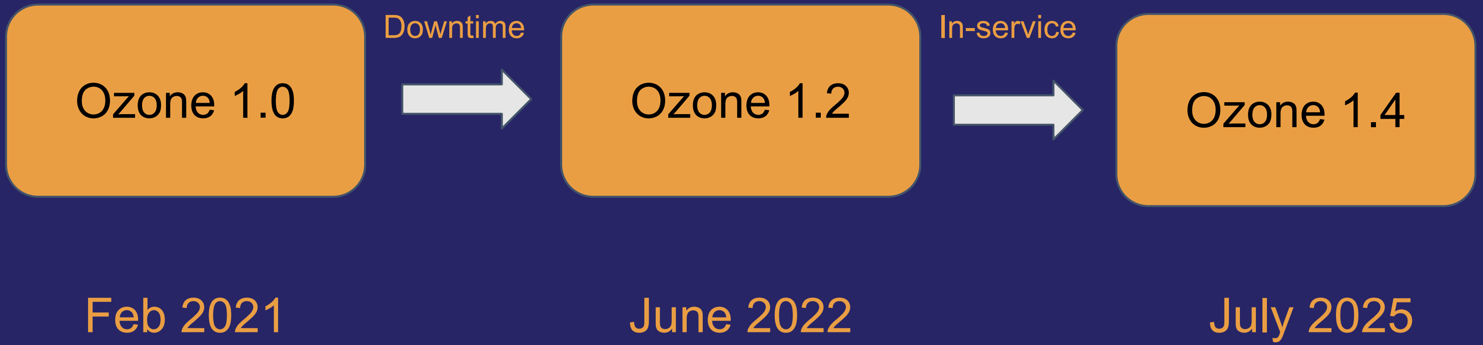
Apache Ozone Best Practices at Shopee

Hui Fei / Shopee

CONTENTS

Overview of Ozone Usage at Shopee

Overview of Ozone Usage at Shopee



Overview of Ozone Usage at Shopee

- Code Maintenance
    - Based on the latest stable branch from the community
    - Build proprietary internal features
    - Backport fixes and improvements from the community
- Internal Features
    - Authentication, Authorization
    - Traffic control on S3G
    - Lifecycle Management
    - Storage class
    - …

COMMUNITY
THE ASF CONFERENCE
CODE

Overview of Ozone Usage at Shopee

- Scale
  - Clusters are located in different countries and across multiple IDCs
  - Over 100 nodes are deployed
  - There are 3 OM services (3 nodes per service)correspond to one SCM service (3 nodes per service) in the main cluster
  - S3G provides a unified global interface
- Storage
  - Billions of objects are stored in total
  - Over 10 PB of storage has been utilized

COMMUNITY OVER CODE THE ASF CONFERENCE

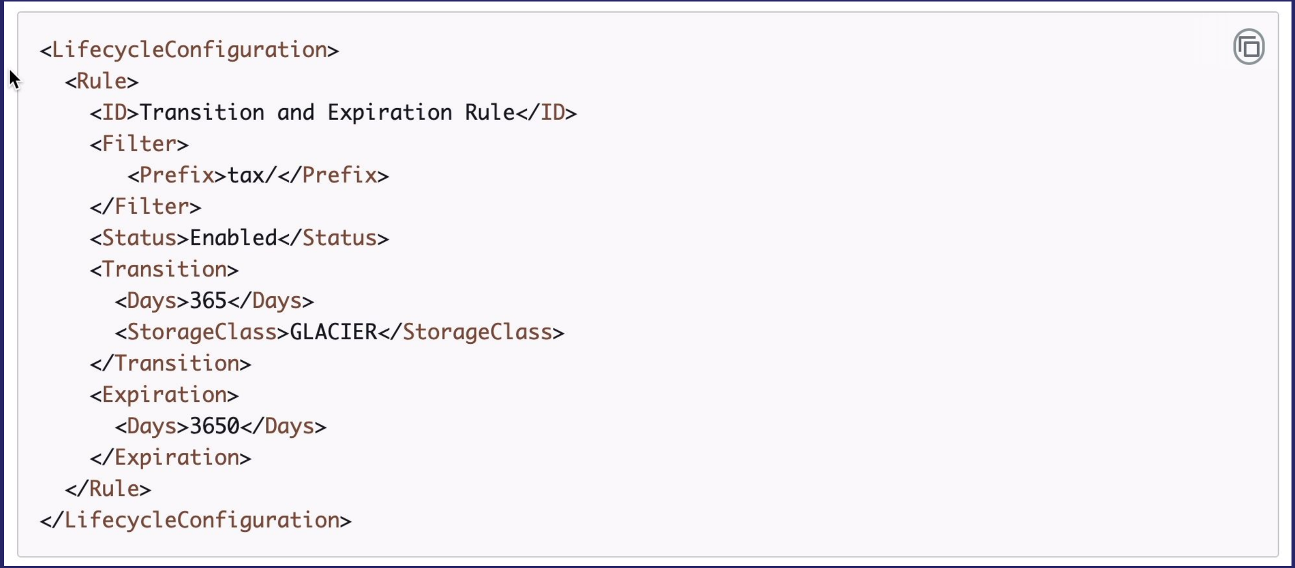
Object Lifecycle Management

Object Lifecycle Management

- Background
  - Too many unused objects need to be cleaned up
  - Users ask the administrator to help clean up unused objects
- Solution
  - An object lifecycle management service
  - It can delete unused objects based on rules set by users
  - The feature should also support AWS features like tagging

Object Lifecycle Management

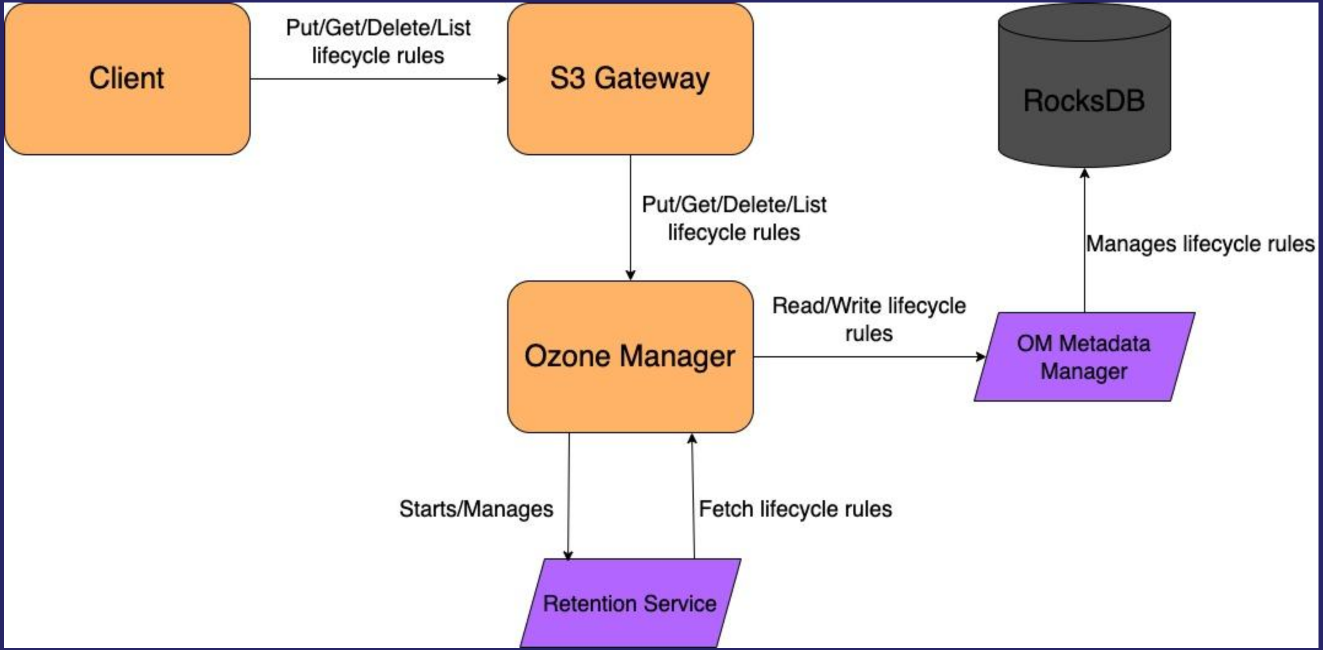
- AWS S3 Lifecycle configurations



```xml
<LifecycleConfiguration>
  <Rule>
    <ID>Transition and Expiration Rule</ID>
    <Filter>
       <Prefix>tax/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>365</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
    <Expiration>
      <Days>3650</Days>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```

Object Lifecycle Management

- Design

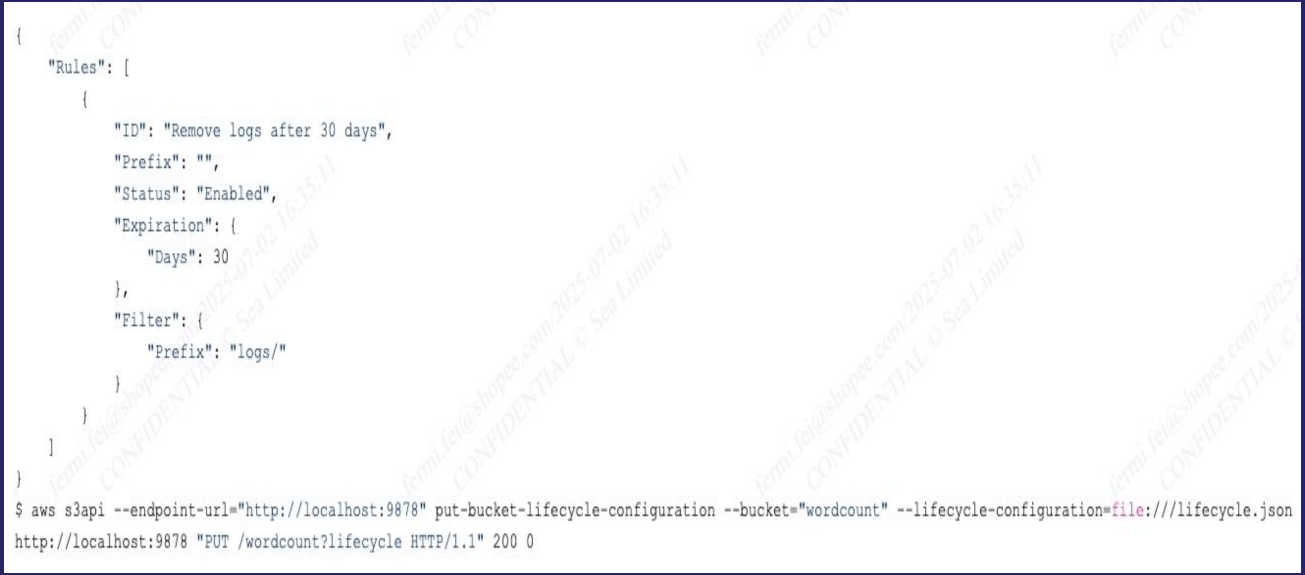


Object Lifecycle Management

- Lifecycle rule
  - id, prefix, enabled, expiration, filter
  - Bind to a bucket
- Comunication protocol
  - Between Ozone client and OM
  - Between AWS client and S3 Gateway
- Retention service
  - Lifecycle table in RocksDB
  - Process rules periodically

Object Lifecycle Management

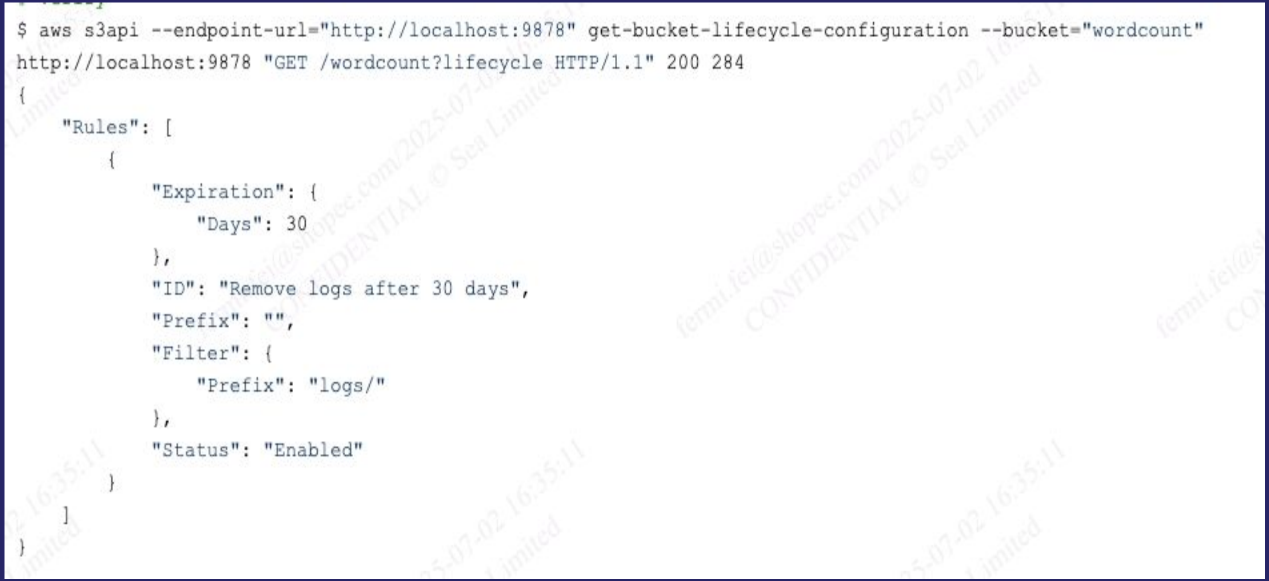
- Example - put-bucket-lifecycle-configuration



```
{
    "Rules": [
        {
            "ID": "Remove logs after 30 days",
            "Prefix": "",
            "Status": "Enabled",
            "Expiration": {
                "Days": 30
            },
            "Filter": {
                "Prefix": "logs/"
            }
        }
    ]
}
$ aws s3api --endpoint-url="http://localhost:9878" put-bucket-lifecycle-configuration --bucket="wordcount" --lifecycle-configuration=file:///lifecycle.json
http://localhost:9878 "PUT /wordcount?lifecycle HTTP/1.1" 200 0
```

Object Lifecycle Management

- Example - get-bucket-lifecycle-configuration



```
$ aws s3api --endpoint-url="http://localhost:9878" get-bucket-lifecycle-configuration --bucket="wordcount"
http://localhost:9878 "GET /wordcount?lifecycle HTTP/1.1" 200 284
{
    "Rules": [
        {
            "Expiration": {
                "Days": 30
            },
            "ID": "Remove logs after 30 days",
            "Prefix": "",
            "Filter": {
                "Prefix": "logs/"
            },
            "Status": "Enabled"
        }
    ]
}
```

COMMUNITY THE ASF CONFERENCE CODE

Object Lifecycle Management

- Rollout at Shopee
  - Released Since June 2023
  - Launched together with tag support later
  - User feedback has been favorable

Object Lifecycle Management

- Contribute it to the community
    - HDDS-8342 - S3 Lifecycle Configurations - Object Expiration
    - The proposal has been accepted by the community
    - The development work is ongoing

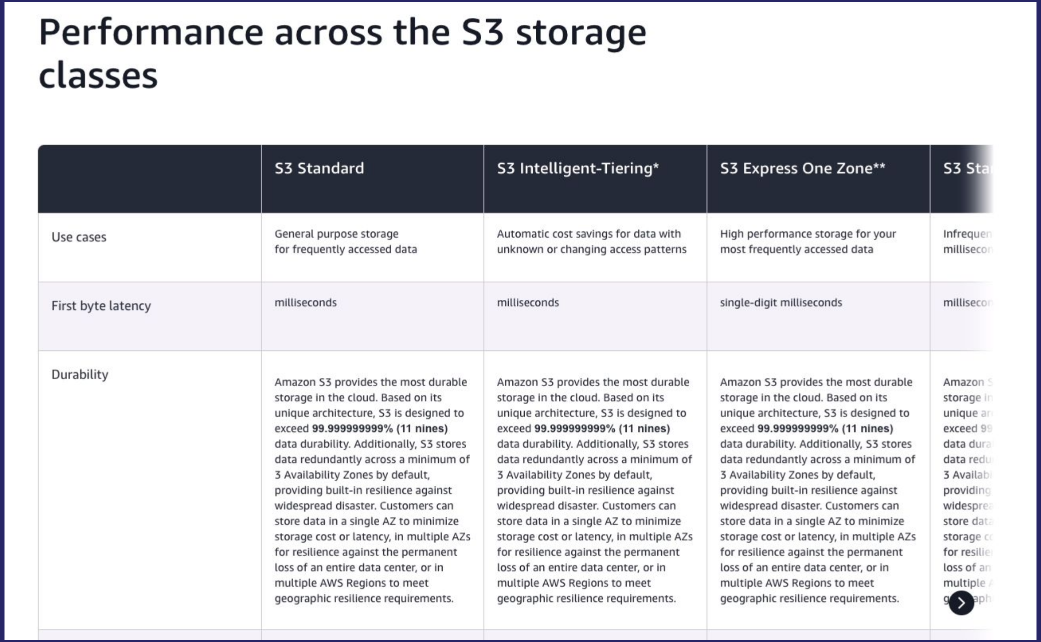
Storage Classes

Storage Classes

- Background
  - Performance requirements for hot data
  - Cost-saving requirements for cold data

Storage Classes

- AWS storage classes



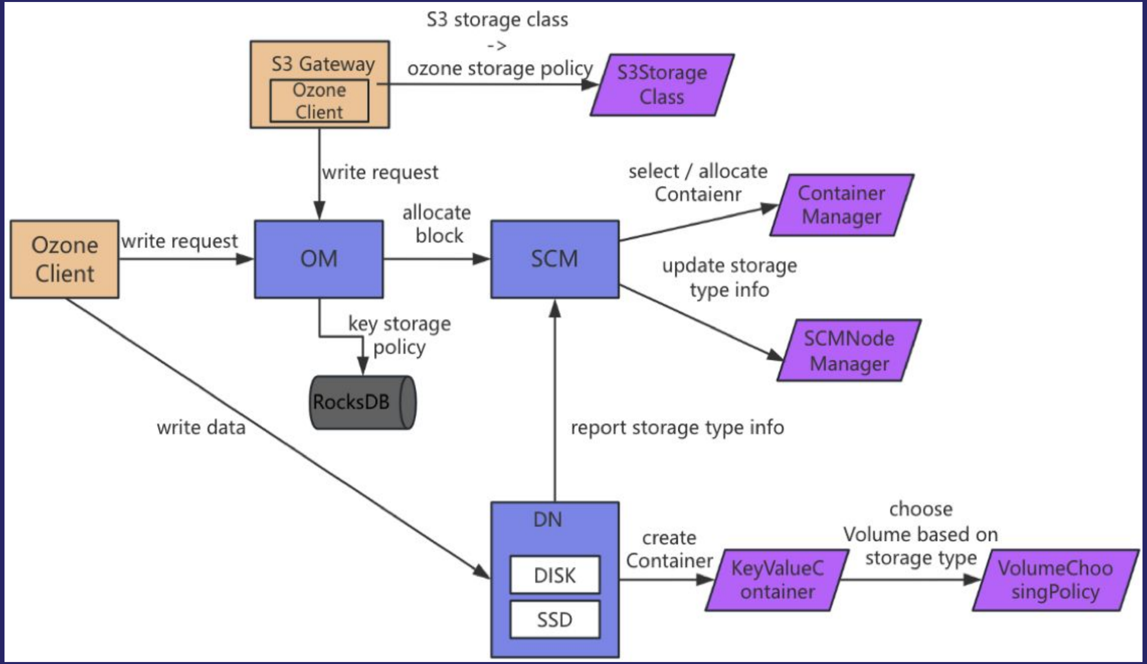
Performance across the S3 storage classes

	S3 Standard	S3 Intelligent-Tiering*	S3 Express One Zone**	S3 Sta
Use cases	General purpose storage for frequently accessed data	Automatic cost savings for data with unknown or changing access patterns	High performance storage for your most frequently accessed data	Infrequen millisecon
First byte latency	milliseconds	milliseconds	single-digit milliseconds	millisecon
Durability	Amazon S3 provides the most durable storage in the cloud. Based on its unique architecture, S3 is designed to exceed **99.999999999% (11 nines)** data durability. Additionally, S3 stores data redundantly across a minimum of 3 Availability Zones by default, providing built-in resilience against widespread disaster. Customers can store data in a single AZ to minimize storage cost or latency, in multiple AZs for resilience against the permanent loss of an entire data center, or in multiple AWS Regions to meet geographic resilience requirements.	Amazon S3 provides the most durable storage in the cloud. Based on its unique architecture, S3 is designed to exceed **99.999999999% (11 nines)** data durability. Additionally, S3 stores data redundantly across a minimum of 3 Availability Zones by default, providing built-in resilience against widespread disaster. Customers can store data in a single AZ to minimize storage cost or latency, in multiple AZs for resilience against the permanent loss of an entire data center, or in multiple AWS Regions to meet geographic resilience requirements.	Amazon S3 provides the most durable storage in the cloud. Based on its unique architecture, S3 is designed to exceed **99.999999999% (11 nines)** data durability. Additionally, S3 stores data redundantly across a minimum of 3 Availability Zones by default, providing built-in resilience against widespread disaster. Customers can store data in a single AZ to minimize storage cost or latency, in multiple AZs for resilience against the permanent loss of an entire data center, or in multiple AWS Regions to meet geographic resilience requirements.	Amazon storage unique ar exceed 99 data dura data redu 3 Availab providing widesprea store data storage for resilie loss of an multiple [illegible]

COMMUNITY THE ASF CONFERENCE CODE

The architecture for new data

- OM
- SCM
- DataNode
- Ozone Client
- AWS Client
- S3 Gateway



COMMUNITY THE ASF CONFERENCE CODE

Storage Classes

- Mapping between AWS storage classes and Ozone policies.

AWS S3 Storage Classes	Ozone StoragePolicy
STANDARD	Hot
STANDARD_IA*	Warm
GLACIER	COLD

- Mapping between Storage PoliciesTo volume storage type

Storage Policy	Storage Tier for Write	Fallback Tier for Write
Hot	SSD	DISK
Warm	DISK	<none>
Cold	ARCHIVE	<none>

COMMUNITY THE ASF CONFERENCE CODE

Storage Classes

Write key to SSD by S3 command

- aws s3api put-object --bucket bucket1 --key key1 --body localKey1 --storage-class STANDARD_IA --endpoint http://xxxx

Write key to SSD by ofs command

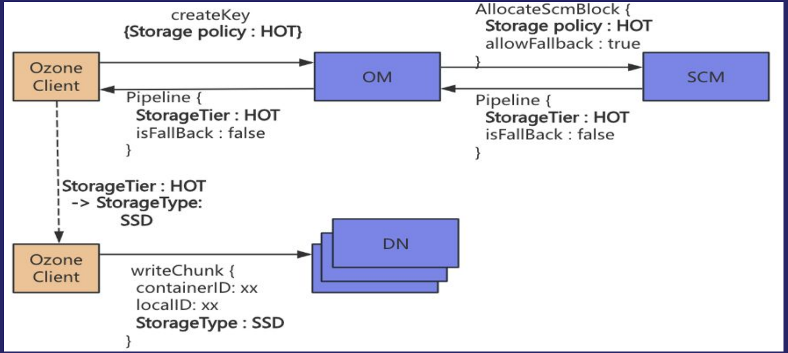
- ozone sh key put s3v/bucket1/key1 localKey1 -sp HOT

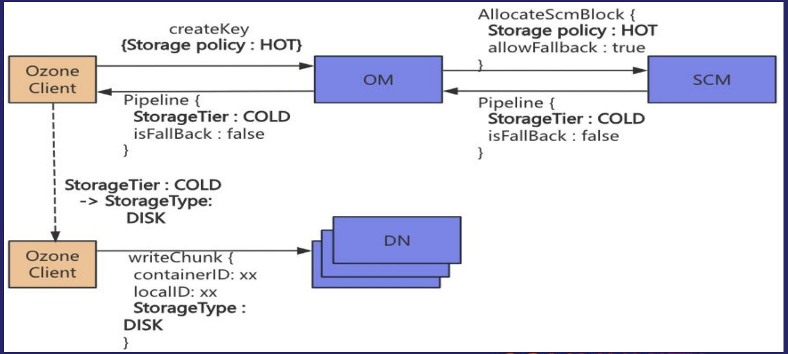
Write key to SSD by API

- createKey (String volumeName, String bucketName, String keyName, …, StoragePolicy storagePolicy)
- reWriteKey (String volumeName, String bucketName, String keyName, …, StoragePolicy storagePolicy)
- createStreamKey (String volumeName, String bucketName, String keyName, …, StoragePolicy storagePolicy)
- createFile (String volumeName, String bucketName, String keyName, …, StoragePolicy storagePolicy)
- createStreamFile (String volumeName, String bucketName, String keyName, …, StoragePolicy storagePolicy)
- initiateMultipartUpload (String volumeName, String bucketName, String keyName, …, StoragePolicy storagePolicy)

Storage Classes

- Normal write (no fallback):
  - Client sends CreateKey request to OM with HOT
  - OM requests SCM to allocate Container with HOT
  - SCM allocates Container and Pipeline with HOT
  - Client gets the mapping from HOT to SSD and writes to DNs
  - DNs in the pipeline create Containers or writes Chunks to SSD Disks
- Fallback write:
  - Client sends CreateKey request to OM with HOT
  - OM requests SCM to allocate Container with HOT
  - SCM allocates Container and Pipeline with HOT. Will fallback to WARM or COLD if no HOT
  - Client gets the mapping from HOT to SSD and writes to DNs
  - DNs in the pipeline creates Containers or writes Chunk to SSD Disks

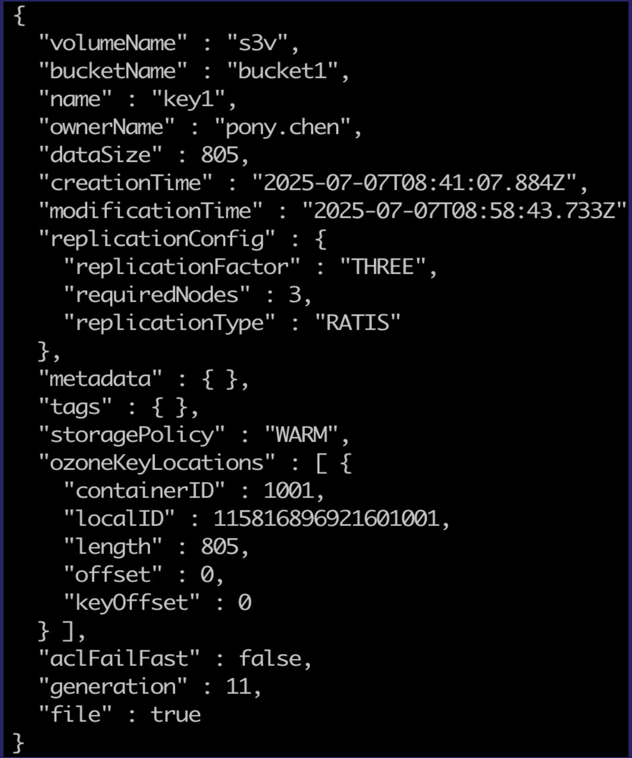




COMMUNITY OVER CODE
THE ASF CONFERENCE

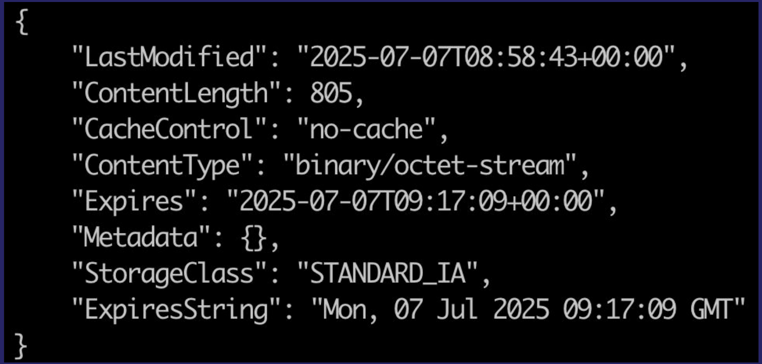
Storage Classes

ozone sh key info s3v/bucket1/key1



```
{
  "volumeName" : "s3v",
  "bucketName" : "bucket1",
  "name" : "key1",
  "ownerName" : "pony.chen",
  "dataSize" : 805,
  "creationTime" : "2025-07-07T08:41:07.884Z",
  "modificationTime" : "2025-07-07T08:58:43.733Z"
  "replicationConfig" : {
    "replicationFactor" : "THREE",
    "requiredNodes" : 3,
    "replicationType" : "RATIS"
  },
  "metadata" : { },
  "tags" : { },
  "storagePolicy" : "WARM",
  "ozoneKeyLocations" : [ {
    "containerID" : 1001,
    "localID" : 115816896921601001,
    "length" : 805,
    "offset" : 0,
    "keyOffset" : 0
  } ],
  "aclFailFast" : false,
  "generation" : 11,
  "file" : true
}
```

aws s3api head-object --bucket bucket1 --key key1 --endpoint xxx

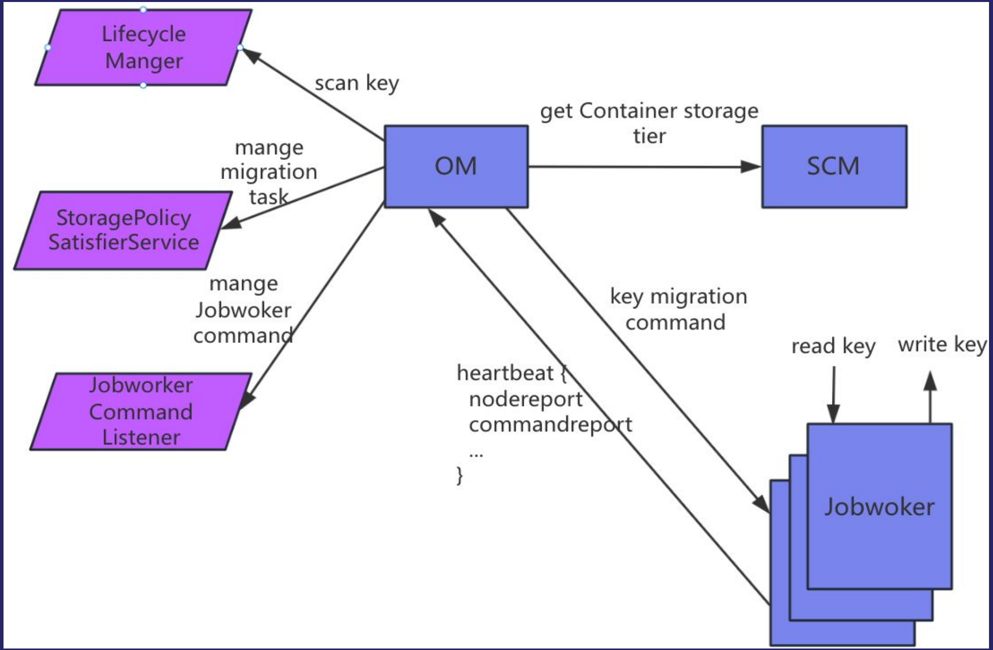


```
{
    "LastModified": "2025-07-07T08:58:43+00:00",
    "ContentLength": 805,
    "CacheControl": "no-cache",
    "ContentType": "binary/octet-stream",
    "Expires": "2025-07-07T09:17:09+00:00",
    "Metadata": {},
    "StorageClass": "STANDARD_IA",
    "ExpiresString": "Mon, 07 Jul 2025 09:17:09 GMT"
}
```

COMMUNITY THE ASF CONFERENCE CODE

The architecture for existing data

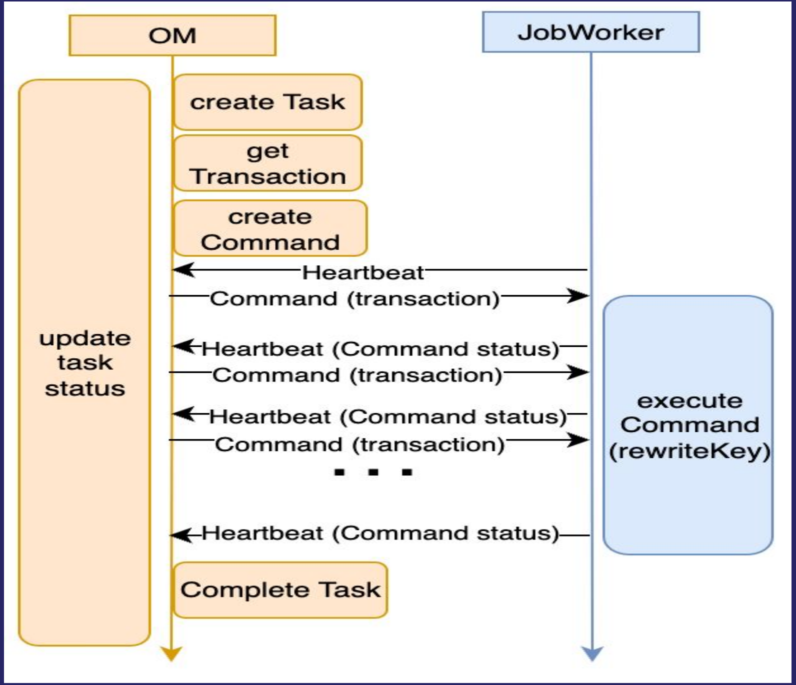
- OM
  - Lifecycle Manager
  - Storage Policy SatisfierService
  - Jobworker Command Listener
- SCM
- JobWorker


Lifecycle Manger
scan key
get Container storage tier
OM
SCM
mange migration task
StoragePolicy SatisfierService
mange Jobwoker command
key migration command
read key
write key
heartbeat { nodereport commandreport ... }
Jobworker Command Listener
Jobwoker

COMMUNITY OVER CODE
THE ASF CONFERENCE

Storage Classes

- Lifecycle manager creates Migration Task on OM side based on the storage policy
- StoragePolicy SatisfierService obtains Migration Task and encapsulates them into command and sends them to Jobworker
- Jobworker uses rewriteKey interface to rewrite Key atomically, and report the command execution result to OM through heartbeat
- OM updates the status of Migration Task according to the execution result



COMMUNITY OVER CODE
THE ASF CONFERENCE

Storage Classes

- Rollout at Shopee
  - Release in progress
  - Will enable the EC to save storage

Storage Classes

- Contribute it to the community
  - HDDS-11233 - Ozone Storage Policy Support
  - Will propose detailed design documentation

Future plans

Future plans

- Improve throughput of Ozone cluster
- Enable EC to save storage
- Enhance Ozone S3-related features
- Bucket synchronization across clusters
- Disaster Recovery Plan for Ozone Cluster DC

COMMUNITY
THE ASF CONFERENCE
CODE

Thanks

Hui Fei