COHESITY

CloudArchive - Best Practices Guide

Version 2.0

August 2025

ABSTRACT

This guide provides an overview of Cohesity CloudArchive, the best practices, and recommendations for using the CloudArchive solution. Best practices for object-based tape are not covered in this guide. The features and functionalities outlined in this CloudArchive Best Practices document are up to the 7.2 Release. Users should be aware that future releases may introduce changes to the described features, functionalities, or behaviors. As such, there is no guarantee that the information presented here will remain consistent or applicable in subsequent software versions.

We recommend that users refer to the most up-to-date documentation to ensure they work with accurate and relevant information.

Table of Contents

Figures

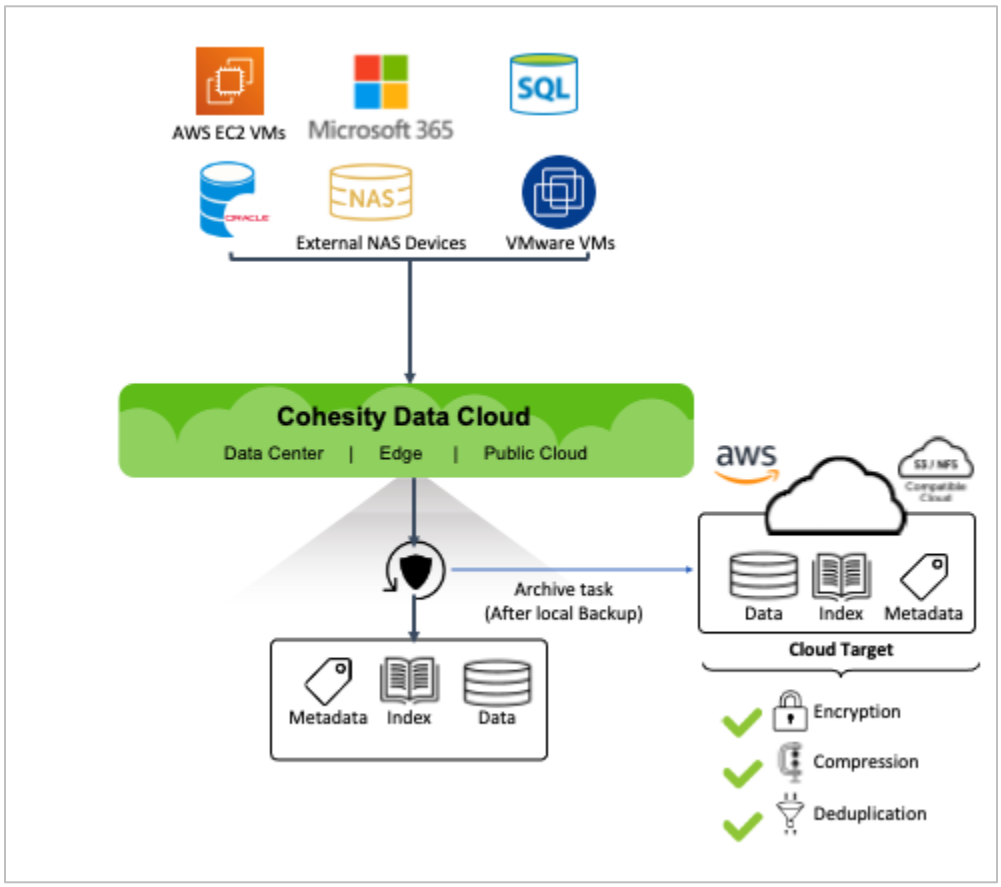
CloudArchive Introduction

Long-term data and application retention is critical to organizations that seek to prevent data loss and meet security, legal, and compliance requirements. The exponential growth of data volumes and the resulting IT management demands have prompted businesses to seek out more cost-effective, reliable data storage and protection solutions.

With Cohesity, IT organizations save time by quickly archiving data to multiple targets — public clouds, private clouds, and any S3-compatible device, while increasing operational efficiency and lowering the total cost of ownership (TCO).

Cohesity's CloudArchive (CA) brings data protection and recovery together in a single coherent solution, both on-premises and in the cloud.

Figure 1: Cohesity CloudArchive



CloudArchive Versions

Cohesity CloudArchive has two versions. Review the differences between the versions and the supported sources before you select an archival method:

CloudArchive Incremental with Periodic Full	CloudArchive Incremental Forever — available from 6.6.0a onwards	CloudArchive – Always Full
Incremental with periodic full (Data and metadata need a full upload every 90 days)	Incremental Forever (does not require any periodic full uploads)	CloudArchive to Object-based Tape is Always Full (No Incremental uploads)
Deduplication is at the Protection Group level	Deduplication at the Storage Domain level	No Deduplication

NOTE: If the sources also support CloudArchive Incremental Forever archival format, Cohesity recommends using CloudArchive Incremental Forever archival format to reduce overall costs for storing the CloudArchive data.

Archival Considerations

- Cohesity recommends archiving data in Incremental Forever archival (CloudArchive Incremental Forever) format. This removes the periodic full backup requirement and provides cost-effective backup.
- To accomplish maximum storage efficiency with CloudArchive Incremental Forever, Cohesity recommends using a single bucket rather than multiple buckets. CloudArchive Incremental Forever deduplicates data at the storage domain level. It provides greater efficiency if all data is archived to the same external target and simplifies the management of external targets, thus reducing cloud costs.
- Cohesity recommends using multiple buckets and Storage Domains if the workloads and associated storage efficiencies align with different storage efficiency settings.
  - For example, if a database such as Oracle TDE is already encrypted, encrypting it again when storing the data in Cohesity storage or an external target is unnecessary. Therefore, you may disable encryption at the Storage Domain level in these situations.
    - If you disable encryption only at the external target level while the Storage Domain level encryption is still active, Cohesity will encrypt the data when storing it on Cohesity storage. Then it will have to decrypt it again when writing to the external target.
- For workloads with different storage efficiency settings, Cohesity recommends archiving them to separate external targets under different Storage Domains with the efficiencies either enabled or disabled based on the source workload.
- You can also enable or disable storage efficiency features at the target level.
  - For example, Oracle databases that are encrypted provide better compression and do not offer many deduplication benefits. CloudArchive Incremental with Periodic Full allows you to turn off source-side deduplication and enable compression.
  - Please note that you cannot turn off source-side deduplication in CloudArchive Incremental Forever.
  - Re-registering the same external target with different configuration settings is not recommended, as it will overwrite the previous settings. To ensure optimal efficiency in such situations, Cohesity recommends using distinct buckets for archiving purposes.
- You can add multiple archival schedules that use the same or different External Targets and the same or different intervals and retention periods to a given Protection Policy. When you add more schedules and send them to the same External Target with different retention and schedule times, the schedules rationalize among themselves. Only the necessary archive is sent, with the longest retention. For example, if you add these three archival schedules to the same External Target:
  - 1. Once a day, retain for 90 days
  - 2. Once every 7 days, retain for 180 days
  - 3. Once every 30 days, retain for 365 days

- Then, on day 7, only one archive is sent, meeting both Schedule 1 and Schedule 2 (and retained for 180 days, per Schedule 2, as it is the longer of the two). On Day 30, only one archive is sent, meeting both Schedule 1 and Schedule 3, but it is retained for 365 days to meet the Schedule 3 retention requirement.

  NOTE: With CloudArchive Incremental with Periodic Full format, the retention schedule will hold the snapshots for a longer period because of the chain deduplication settings.

- By contrast, if you send the archives to different External Targets, then:
  - On Day 7, per Schedule 1, the archive is sent to the first External Target and retained for 90 days.
  - Per Schedule 2, the archive is also sent to the second External Target and retained for 180 days.
  - On Day 30, per Schedule 1, the archive is sent to the first External Target and retained for 90 days.
  - Per Schedule 3, the archive is also sent to the third External Target and retained for 365 days.

  When you use multiple schedules with different External Targets, the schedules don't rationalize, and you accrue network and storage usage for each scheduled run.

- CloudArchive Incremental with Periodic Full archives data using multipart uploads.
- With multipart uploads, large objects are created, with an average object size of 250GB. The PUT request would be 32MB in size.
- CloudArchive Incremental Forever archival format does NOT use multipart upload.
- By avoiding the overhead associated with multipart uploads, where data is divided into parts and individually uploaded, smaller files or chunks of data from different files are grouped to form a logical bundle in the range of 8MB-24MB with an average size of 16MB.
- Cohesity supports any S3-compatible device as an S3-Standard tier (Regular as storage class) external target with similar capabilities to an AWS S3-Standard.
- For Cloud Archive with incremental forever and for CloudArchive Direct, Cohesity cluster may download a portion of the data from the S3-compatible and NAS external targets during each garbage collection process and reupload it after compaction. This can potentially lead to increased network bandwidth usage. To prevent additional network usage, Cohesity recommends using S3-compatible and NAS targets hosted within the same data center or network as the Cohesity cluster.
- Do not apply an AWS Object Lifecycle Policy with the Expiration action to an AWS S3 Compatible External Target used for archiving data. Applying a policy with this action may result in data loss.
- You can register an external target either for Cloud Archive, CloudArchive Direct, or for Cloud Tier. You cannot register the same external target for Cloud Archive, CloudArchive Direct, and Cloud Tier purposes.
- If you are using an S3 bucket with object versioning enabled as the external target, you must set up a Lifecycle Management Policy on the bucket to permanently delete non-current versions of objects. This is required to garbage-collect object versions. If there is a malware attack on the bucket that overwrites object versions, this policy could result in data loss in the absence of WORM locks on the object versions.

- If an external target you want to register for archival has an object lock with a default retention period configured, then you must not select "incremental forever" as the archival format while registering this external target with the Cohesity cluster.
- Cohesity may require additional permissions to validate whether such a data lock policy is configured on the external target. For example, to validate the data lock configuration on AWS S3 targets, you must have the s3:GetBucketObjectLockConfiguration permission.
- If the **Object** Key **Pattern** of the S3 View is **Object ID**, then you must use the **Incremental Forever** archival format while archiving the S3 View. You cannot archive the S3 View using the **Incremental with Periodic Full** format.
- If you are planning to archive your data to Azure hot tier using the incremental forever archival format, then ensure that the storage account you create is accessible through a public network and not a private network.
- To avoid early deletion charges, garbage collection of objects archived to an external target is performed only after the minimum retention period is met. As a result, storage utilization at the target level may increase; however, the associated costs typically remain the same or may even be lower.
- Any S3-Compatible Object-based Tape Storage targets that use tape as its backend must be registered as an Object-based Tape Storage Class.
- The Object-based Tape Storage Class uses S3 Glacier APIs for archival and recovery.
- For archiving data to **S3 Compatible - Tape Based Storage Class**:
  - The archival format is Always Full with compression enabled by default.
  - Cohesity does not support deduplication and incremental archive.
  - Cohesity does not support CloudRetrieve of data archived to S3 Compatible-Tape Based.
  - Cohesity does not support file-level recovery.
  - When a recovery is initiated from the Object-based Tape, the data being recovered is brought into the BlackPearl cache and retained for a default of nine days (cache for seven days plus an extra two). During this time, the cache must also accommodate ongoing daily backup writes.

    NOTE: By default, the recovery workflow retains the restored data from the tape on the S3-compatible cache for nine days. This is to ensure that rehydration of data is successful. If you want to reduce the retention period of restored data on the S3 cache, contact Cohesity Support.
  - If multiple recoveries are triggered within seven days, the cache size must be large enough to hold all the recovered snapshots.
  - By default, the number of parallel streams per cluster is set to 10. If you want to configure these values, contact Cohesity Support.
  - By default, the number of restore tasks per Tape Storage is set to 1.
  - Cohesity recommends not to add more objects to a Protection Group, as the recovery workflow could hydrate the complete archive while restoring a single object.

- For information on the workflows supported for each external target, see Supported Workflows and External Targets.

> **NOTE**: CloudArchive Incremental Forever archival format does not offer customizable parameters to enforce strict minimum or maximum object sizes. While the system targets specific size ranges, the inherent variability in data compressibility precludes guaranteed exact sizes. This design philosophy prioritizes overall performance optimization over rigid size constraints.

Archive Object Lock Considerations

Cohesity stores immutable data at both the source cluster and external target levels. By configuring Object Lock, you can lock archives on external targets to prevent data from being modified, deleted, or overwritten. Object Lock can be configured during external target registration.

- **Confirm Object Lock support**: Verify that the target you select supports Object Lock. For the list of targets that support Object Lock, see Supported Workflows and External Targets.
- **For AWS external targets**: The Object Lock-supported AWS external target must have versioning and Object Lock enabled. For more information, see Using S3 Object Lock.
- **For Azure external targets**: The Object Lock-supported Azure external target must have version-level immutability enabled on the storage container, which indicates that the storage account must also have version-level immutability. For more information, see Enable version-level immutability support on a container.

  NOTE:

  - The above properties must be set at the time of target creation and cannot be enabled after creation.
  - Avoid setting default retention on S3 buckets and Azure - this is not required as Cohesity locks the objects for the duration required. Large default retentions affect garbage collection and increase costs.
  - For AWS and S3-compatible targets, by default, the archived objects will be locked in the Governance retention mode. Support for enabling Object Lock in **Compliance** mode via the Cohesity UI will be introduced in an upcoming release.
  - For Azure targets, by default, the objects will be locked in the Locked mode

- If you registered an Object Lock-capable external target on a Cohesity cluster running version 6.8 or earlier and later upgraded the cluster to a version that supports Object Lock, you will not be able to enable Object Lock on those previously registered targets.
- When archiving data to versioned buckets, ensure that you do not edit the storage class of objects directly. Editing the storage class of objects directly will create a new version of the object while the original version remains in the previous storage tier. Object-level locks will not be carried over to the new version.
- The Cohesity Data Movement feature, which can be enabled during protection policy creation, cannot coexist with Object Lock. An archival target can only have one of the settings enabled.
- The expiry of the object lock will match the expiry set for archived snapshots. However, the object lock will not update if you modify the expiry for archived snapshots after the Protection Group has run.
- You cannot disable Object Lock once you enable it on a target.
- **For archival with periodic full format**:
  - For the list of targets (S3-compatible, AWS, and Azure) that support Object Lock for archival with periodic full, see Supported Workflows and External Targets.
  - If Object Lock is not met with an archive, the archive is still considered successful and can be recovered.

  - After archival, to view the Object Lock status, perform the following:
    - Click the **Protection** menu.
    - Select the required Protection Group.
    - Select the required run.
    - Click the **Cloud Archive** tab.

The **Object Lock Compliance** status and **Archive Object Lock Expiry Time** will be displayed.

- For archival with incremental forever format:
  - For the list of targets (S3-compatible, AWS, and Azure) that support Object Lock for archival with incremental full, see Supported Workflows and External Targets.
  - If Object Lock compliance is not met for an archive, then the archival will fail.
  - For on-prem clusters, Cohesity does not support Object Lock for CloudArchive Direct. However, for AWS and Azure NGCE clusters, where the primary copy of data is down-tiered directly to an external target via CloudArchive Direct, support Object Lock for CloudArchive Direct of primary data.
  - With Object Lock enabled, the storage utilization of external targets may be higher.

Performance Considerations

To ensure optimal performance, follow these best practices and guidelines.

Right-sizing the Cluster for Archival

The archival process involves several steps, including reading data from the source, compressing and deduplicating it, storing the primary copy in the Cohesity cluster, and then writing it to archival storage. These tasks can increase disk I/O and CPU utilization, potentially affecting the cluster's overall performance if not managed effectively. To achieve optimal cluster performance, collaborate with Cohesity to size the cluster adequately to accommodate additional nodes for meeting throughput requirements.

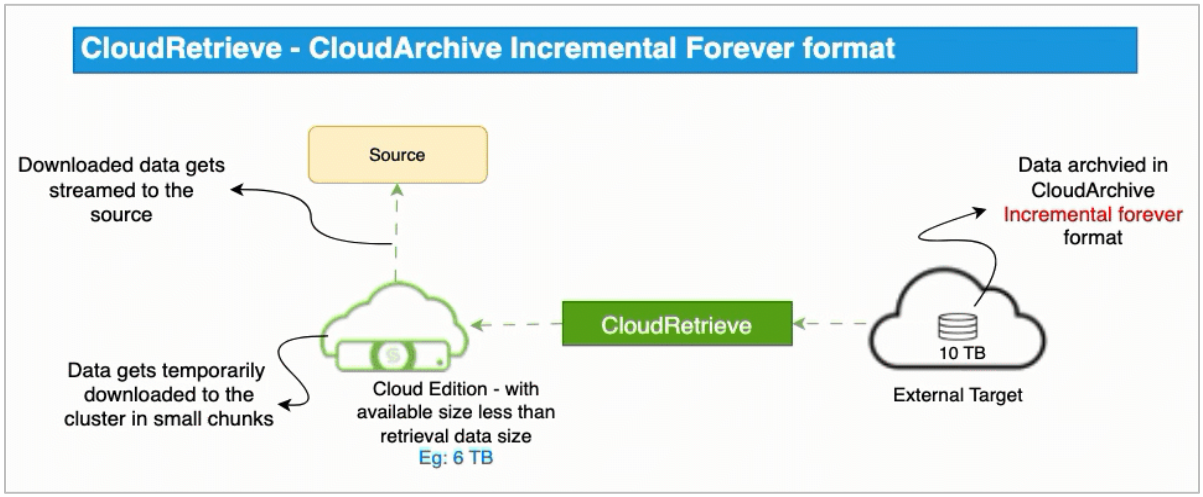
Depending on the archival schedule, the Cohesity Sizer tool will incorporate extra nodes to meet the necessary throughput. It's important to note that the Cohesity Sizer tool will introduce additional throughput requirements only when selecting Daily Archives or more frequent schedules. If Weekly or Monthly archival options are chosen, there will be no adjustments to the requirements, as Cohesity expects the cluster to effectively distribute and manage the archival tasks without adversely affecting cluster performance. However, for guidance on sizing your cluster appropriately for archival purposes and to ensure optimal performance, we strongly recommend contacting your Cohesity Sales Engineer.

- If you intend to implement a shorter archival interval, such as archiving logs every 10 minutes, it will undoubtedly have a noticeable impact on the overall performance of your cluster. The continuous archival operations may lead to conflicts with other backup jobs in the cluster. Cohesity recommends using a dedicated cluster for archival operations in such scenarios.
- Cohesity by default allows 10 streams to perform Archival to S3-Compatible Object-based Tape External Target.
- Contact Cohesity Sales Engineer to assist with sizing the cluster and tuning the cluster to saturate LTO Tapes for Object-based Tape.

Right-sizing the Cluster for Retrieval

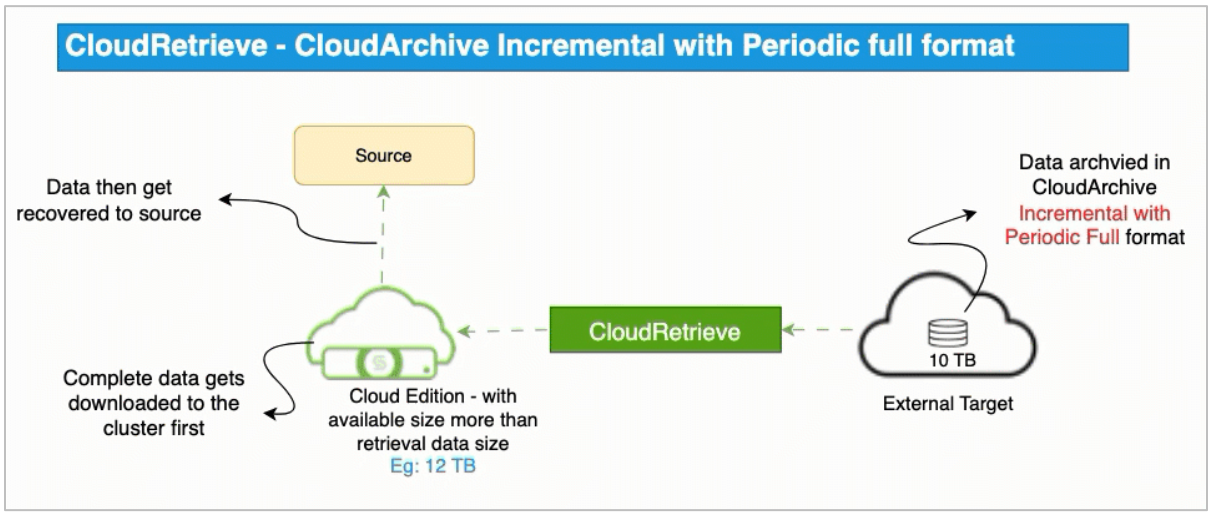
- During a retrieval operation, data is recovered to a new cluster. CloudArchive Incremental Forever archives data directly to the chosen target, whereas CloudArchive Incremental with Periodic Full Format follows a two-step process, first downloading the data to the Cohesity cluster and then retrieving it to the selected target. When retrieving data stored in CloudArchive Incremental Forever archival format, there is no specific cluster size requirement for the retrieval cluster. Cohesity's process involves temporarily downloading the data and recovering it to the chosen source. Therefore, whether you use a smaller or larger (higher storage/compute) cluster, the primary difference lies in the speed of the recovery process.
    - For example, suppose you wish to CloudRetrieve 10TB of data using a new Cohesity Cloud Edition cluster with a size of 6TB. In that case, even though the cluster size is smaller than the amount of data to be recovered, you can still perform the recovery without any issues. However, the recovery speed will be slower, as Cohesity will temporarily download the data and stream it to the requested source based on the available cluster size.

Figure 2: CloudRetrieve - CloudArchive Incremental Forever Format



- When considering a CloudRetrieve operation for data archived in the CloudArchive Incremental with Periodic Full format, confirming that the retrieval cluster possesses sufficient space to accommodate the data you intend to recover is crucial. In this case, Cohesity temporarily downloads the entire dataset onto the retrieval cluster before recovering to the chosen source. Thus, when dealing with CloudArchive in the Incremental with Periodic Full archival format, it is essential to ensure that the retrieval cluster has ample space to accommodate the data download process.
  - For example, if you wish to CloudRetrieve 10TB of data using a new Cohesity Cloud Edition cluster, you must ensure that the cluster has an available size (12 TB) that is more than the retrieval data size. With CloudArchive Incremental with a periodic full archival format, Cohesity first downloads all the data to the retrieval cluster. Then the data is recovered and sent to the source. Unlike CloudArchive Incremental forever format, Cohesity does not stream the downloaded data as it gets downloaded from the external target.

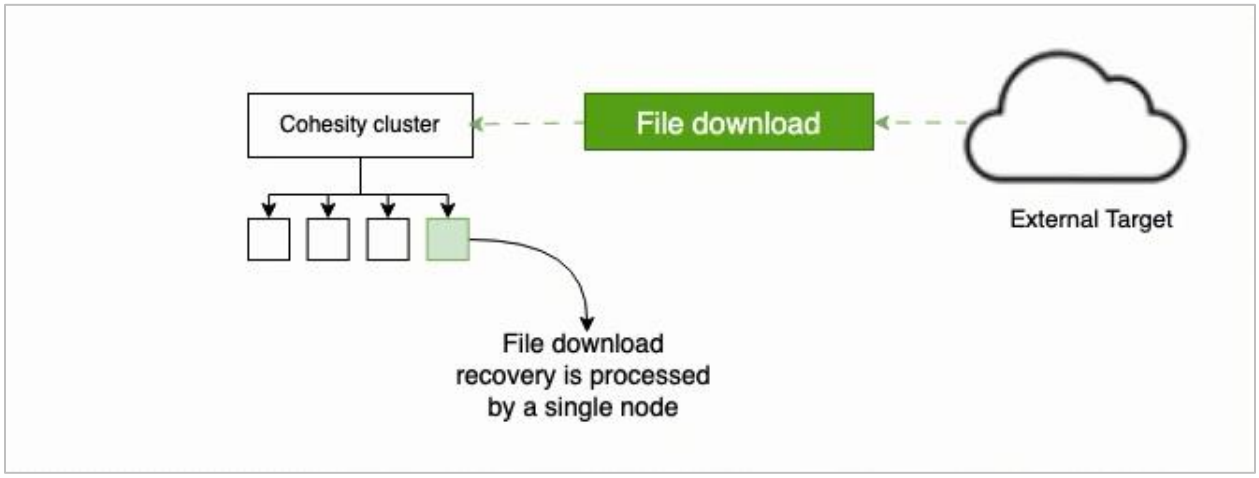
Figure 3: CloudRetrieve - CloudArchive Incremental with Periodic Full Format



File Download Considerations

When initiating a recovery process in Cohesity, you can either download the file or recover it directly to the selected source. Opting to download a large file will inevitably lead to an extended download time, regardless of the number of nodes within your cluster.

Figure 4: Recovery - File Download from CloudArchive



This occurs because:

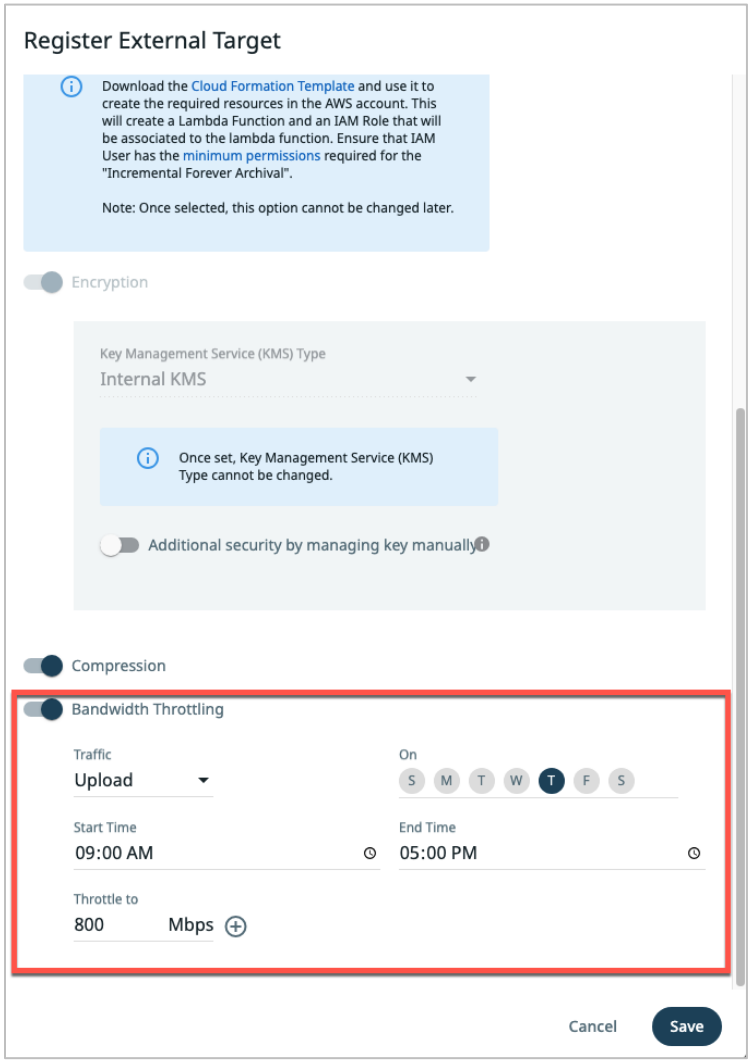
- Cohesity needs to compress the file for downloading.
- The download operation is not distributed across multiple nodes; instead, a single node handles the download operation.

This can adversely affect the overall recovery speed. Therefore, Cohesity recommends performing a file-level recovery to the source instead of downloading large files.

Bandwidth Throttling Considerations

Bandwidth throttling can be a valuable tool for managing bandwidth usage and costs when archiving data to external targets.

Figure 5: CloudArchive - Bandwidth Throttling Configuration



Cohesity suggests implementing bandwidth throttling for external targets exclusively when it's an essential customer requirement. It's crucial to recognize that configuring bandwidth throttling can negatively impact archival and recovery speed.

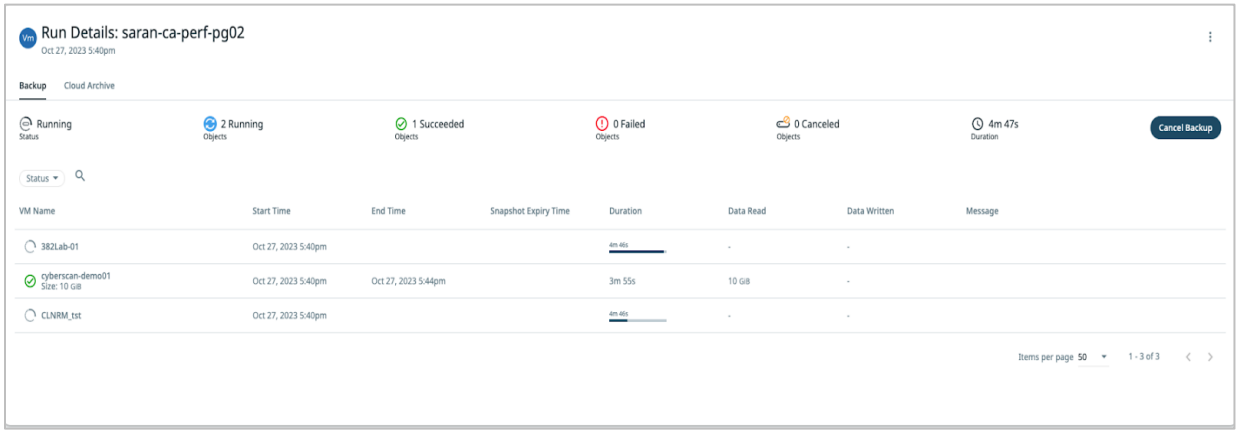
For example, bandwidth throttling can slow down the archival process when archiving a substantial amount of data to S3 over a slow network connection. Similarly, during data recovery from S3 with a slow network connection, bandwidth throttling can significantly extend the time needed for the recovery process. To optimize these processes, consider the following:

- Schedule archival and recovery jobs during off-peak hours to minimize network congestion.
- To expedite the completion of large archival and recovery tasks, break them down into smaller, more manageable jobs.

Streamlining Archival Efficiency Through Protection Group Planning

Cohesity ensures that all local backups are complete before initiating the archival process. If a Protection Group comprises objects of varying sizes, the archival task might be held up until the largest object is successfully backed up. Thus, Cohesity recommends organizing similar-sized workloads within a single Protection Group. This way, the archival process can commence promptly without waiting for the largest object to complete its backup.

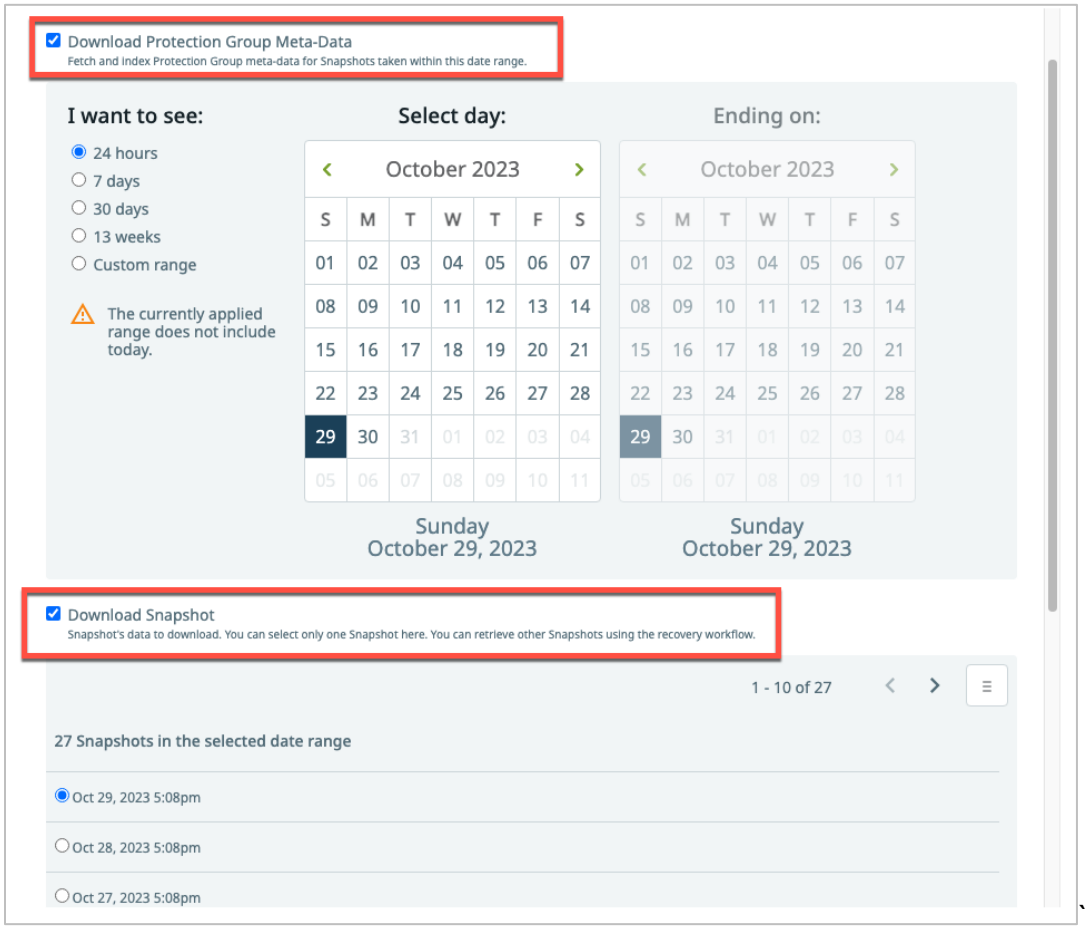
Figure 6: CloudArchive - Archive Jobs Waits for Backup Completion



CloudRetrieve Efficiency: Metadata-First Approach for Swift Data Recovery

During the CloudRetrieve process, once you've chosen a Protection Group, Cohesity will automatically download both the metadata and the most recent snapshot.

Figure 7: CloudRetrieve - Select/Deselect Download Snapshot



If you're uncertain about which snapshot to use for data recovery, it's advisable to opt to download only the metadata. You can then select the appropriate snapshot during the actual recovery workflow. Downloading the snapshot data can be time-consuming, especially when the target is a cold-tier storage option like AWS Glacier or Deep Archive. In such situations, it's more efficient to begin by obtaining the metadata rather than immediately downloading the latest snapshot.

NOTE: For additional performance recommendations, open a Cohesity support ticket to receive expert advice.

NOTE: Cohesity does not support CloudRetrieve of data archived to S3 Compatible-Tape Based.

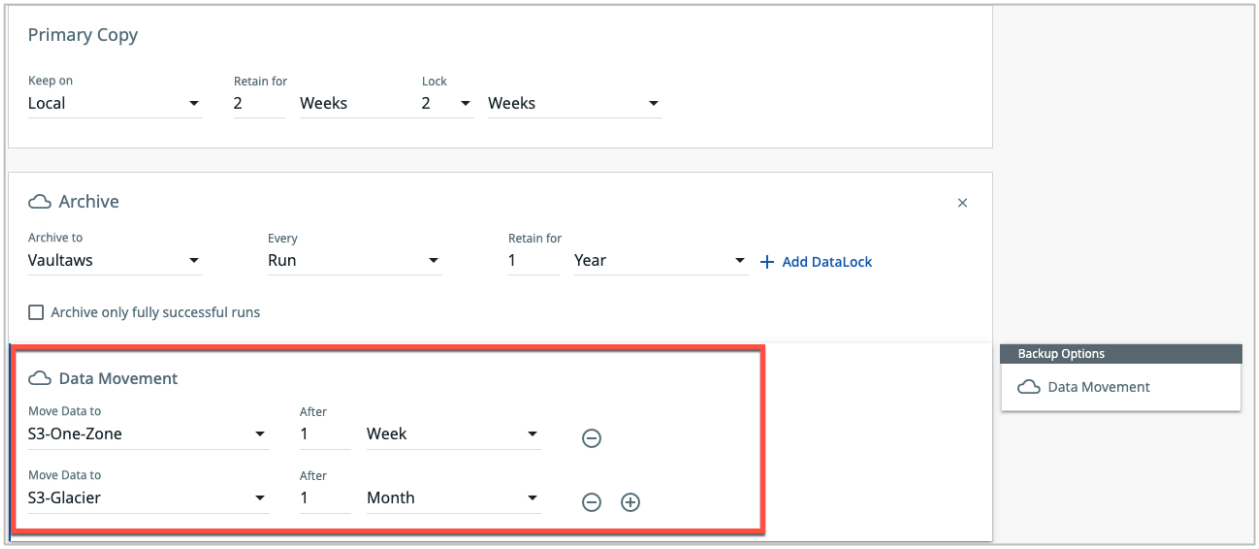
Data Movement Considerations

NOTE: Starting with Cohesity version 7.2.2_u2, the Data Movement feature is deprecated. As best practice, Cohesity no longer recommends using this feature. Instead, it is advised to archive data directly to colder storage tiers (e.g., AWS S3 Glacier or Deep Glacier) based on your archival and retention requirements.

CloudArchive Incremental Forever

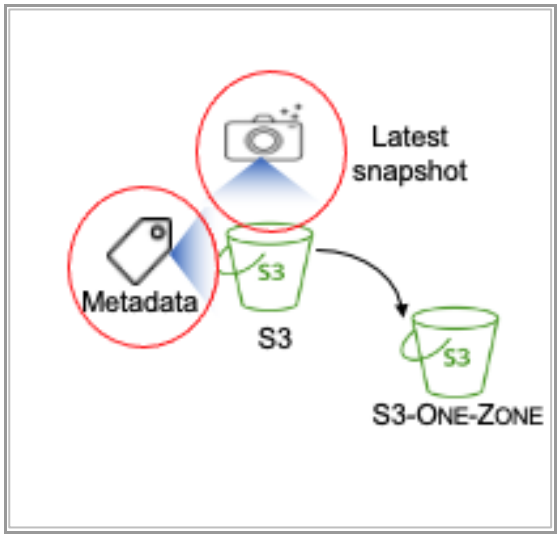
- Cohesity recommends data movement with CloudArchive Incremental Forever (Available from 6.8.1 onwards). The **Data Movement** option allows Cohesity to own the Lifecycle Management (LCM) of data in the external target.
- You can add multiple data movement policies so long as each time the data is being down-tiered to a lower storage class than the previous storage class (in short, uptiering is not supported; data must always be down-tiered).

Figure 8: CloudArchive – Data Movement



- Cohesity keeps all metadata and the latest snapshot in the original/fastest access tier. When Cohesity recovers data from an archive, it first downloads metadata, and having metadata in the higher access tier speeds recovery operations.

Figure 9: CloudArchive – Data Movement to Colder Tiers



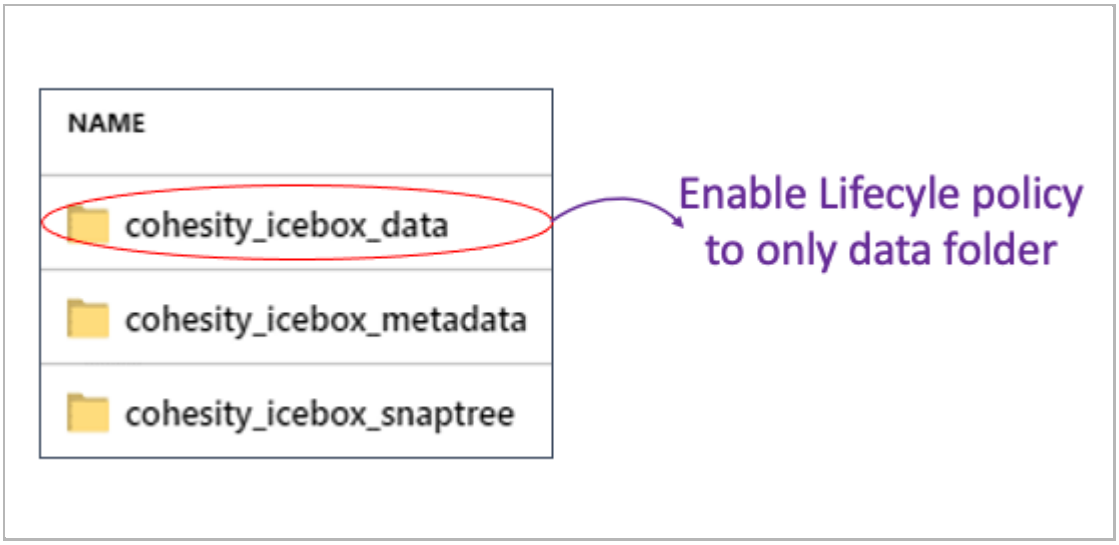
- Cohesity recommends that you plan your recovery service level agreement (SLA) before enabling Data Movement. If you need to recover data often from older snapshots, then using Data Movement is not advisable. Depending on the size of the data and the tier, it could take a significant amount of time to transfer the data from the cold tier to the higher tier for recovery, resulting in a long-running task. It's essential to consider these factors as best practices and plan accordingly.
  - For example, Data Movement is configured from S3 to S3 Glacier, and you need to recover a single file from an older snapshot. In that case, it will take time because the hydration time for S3 Glacier is four hours. Consequently, the time it takes to retrieve the file will depend on its size, resulting in a longer recovery operation.

NOTE: Data Movement operates at the chunk file level, down-tiering only the unreferenced chunk files. On the other hand, CloudArchive Incremental Forever deduplicates data at the bucket level and may involve data being shared between two Protection groups. In this scenario, data will only get down-tiered if it meets all associated down-tiering policies defined. If incremental archival only adds new data without overwriting or deleting existing data in the source, then the latest snapshot will reference all the data, and Cohesity will keep the latest snapshot in the first tier itself, which means there won't be any data that will get down-tiered. To ensure optimal results, verify any changes to the data at the source (including additions, overwrites, and deletions) before enabling Data Movement.

CloudArchive Incremental with Periodic Full

- With CloudArchive Incremental with Periodic Full, customers can use the native Cloud Service Providers (CSP) Lifecycle Management (LCM) to move data between cloud storage tiers to reduce long-term archival costs.
- Cohesity recommends enabling lifecycle policies to data (cohesity_icebox_data) and keeping metadata and SnapTree data in the higher tier. For a restore operation, Cohesity downloads metadata and SnapTree data to determine which data chunks need to be downloaded from the archive. This speeds up the restore operation from a lifecycle policy-enabled bucket.

Figure 10: CloudArchive - Enable CSP LCM



See the AWS documentation to learn how to set up LCM on the bucket.

- Note that the option to tier the "cohesity_icebox_data" folder is unavailable in Azure. Azure will tier everything written in the bucket and doesn't provide any options to select a specific folder for tiering.
- Data recovery from CSP LCM-enabled buckets may take longer than expected. This is due to Cohesity's lack of awareness of data movement managed by the service provider through LCM. During a recovery operation, Cohesity attempts to retrieve the data from the first tier, which is already down-tiered by CSP's LCM. This retrieval operation may take a longer time to complete. It's important to note that Cohesity doesn't support FLR from CSP LCM-enabled buckets. This is because reconstructing FLR data is more time-consuming than performing full recovery, and due to the long-running operation, Cohesity will fail FLR from CSP LCM-enabled buckets.

Migration Considerations

To ensure a smooth migration of CloudArchive Incremental with Periodic Full archives to CloudArchive Incremental Forever format, follow these best practices and guidelines:

- Ensure that the workload supports CloudArchive Incremental Forever format from the support matrix.
- Ensure that the external target configured for CloudArchive Incremental with Periodic Full archival also supports CloudArchive Incremental Forever from the support matrix.
- Cohesity recommends utilizing the S3 protocol for archiving with CloudArchive Incremental forever if the external target supports both NFS and S3 protocols, as it can deliver better performance.
- Cohesity recommends using the existing external target. Once you prepare the external target for CloudArchive Incremental Forever, the jobs will automatically upload a new reference full in CloudArchive Incremental Forever format during the next scheduled periodic reference full upload. CloudArchive continues to archive data in CloudArchive Incremental with Periodic Full format till the following scheduled reference full backup. The old CloudArchive Incremental with Periodic Full jobs will get garbage collected based on the retention policies.
- After migrating the jobs to CloudArchive Incremental Forever, you may notice that compared to CloudArchive Incremental with Periodic Full, there is a relatively higher API cost for CloudArchive Incremental Forever. This is because CloudArchive Incremental Forever writes data in small chunks for efficient space reclamation.
- Cohesity recommends using CloudArchive Incremental Forever, as even with relatively higher API cost, in the long run, you will still save more cost with CloudArchive Incremental Forever because of its efficient space reclamation and deduplication features.

NOTE: If you want to immediately migrate CloudArchive Incremental with Periodic Full jobs to CloudArchive Incremental Forever, contact support to perform a planned (expedited) migration.

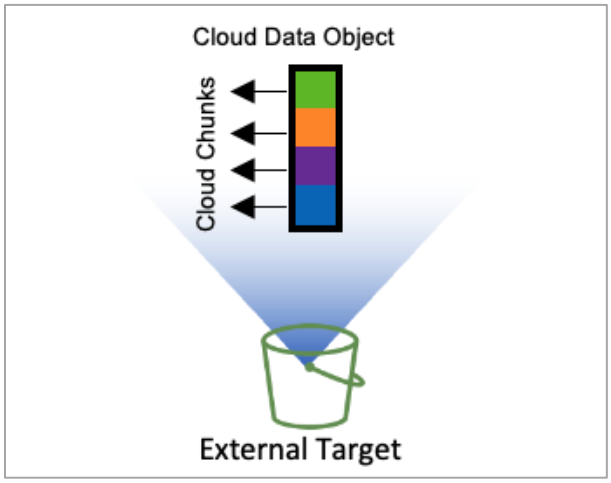
For more information, see the CloudArchive Migration guide.

Garbage Collection Considerations

Cohesity CloudArchive is designed to ensure efficient data management and storage optimization in archival targets. Cohesity performs periodic garbage collection (GC) to efficiently reclaim storage space on external targets such as AWS S3, Azure Blob Storage, and S3-compatible storage systems as part of its comprehensive data archiving solution.

Cloud Data Object Structure

Figure 11: Cloud Data Object Structure



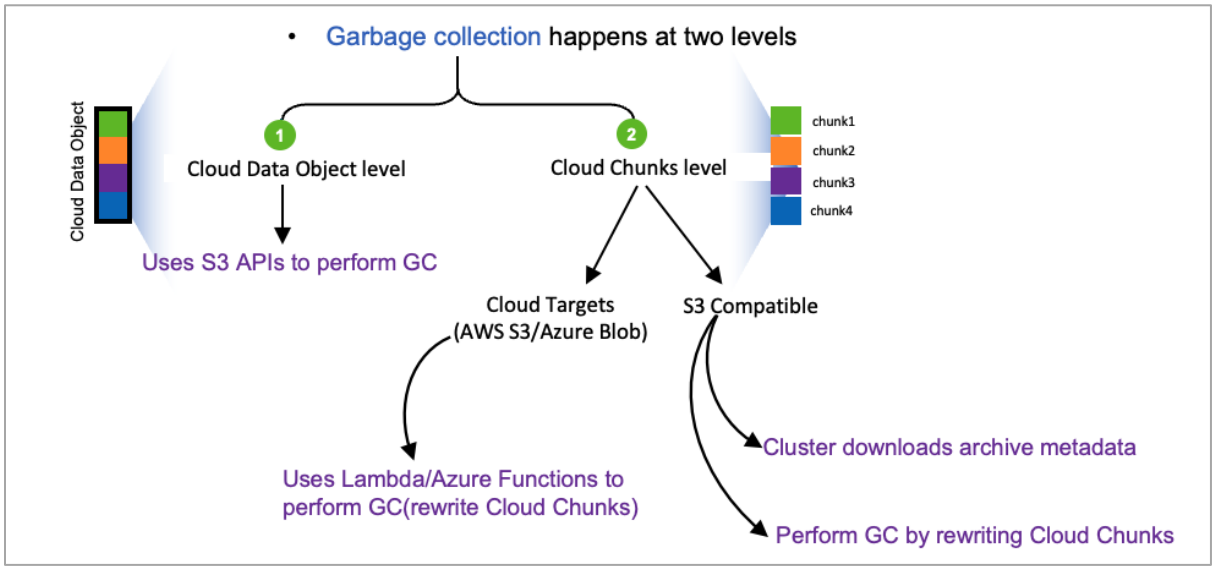
Cohesity writes data to external targets at the Cloud Data Object level. These objects are written in fixed sizes, typically ranging from 4 MB to 32 MB, with an average size of 8MB to 16MB, depending on the size of the archived data. Within each Cloud Data Object, the data is segmented into smaller units known as cloud chunks, which vary in size, generally between 8 KB and 16 KB.

To optimize storage and efficiency, Cohesity attempts to group approximately 1,000 chunks into a single chunk file. The capacity of each chunk file varies based on the chunk size; for example, an 8 MB chunk file can store up to 1,000 chunks of 8 KB each. However, if the chunk size increases (e.g., to 16 KB), fewer chunks can be stored within a chunk file of the same size.

Garbage Collection Process

The Cohesity cluster automatically manages garbage collection and occurs at two levels:

Figure 12: Garbage Collection Process



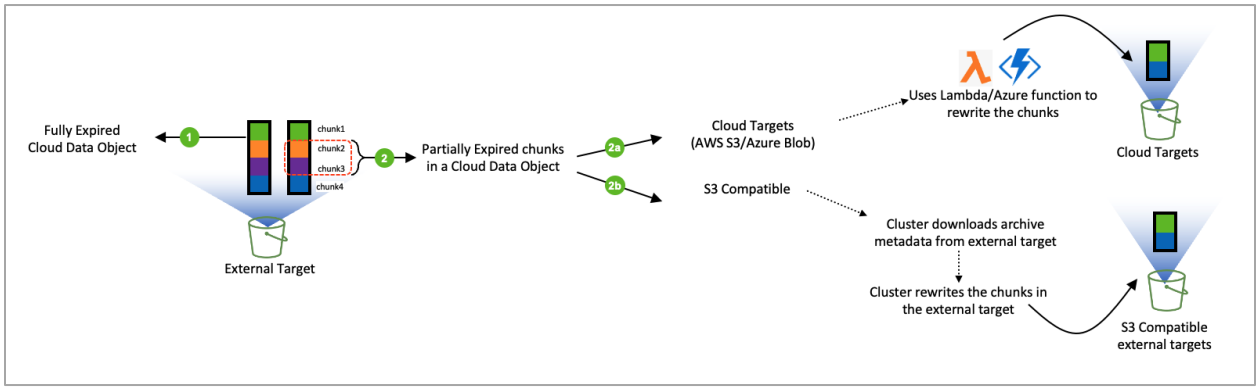
Cloud Data Object Level

When an entire Cloud Data Object is identified as expired, Cohesity uses the appropriate cloud provider's API (e.g., AWS S3 or Azure Blob Storage) to delete the object, thereby reclaiming the associated storage space.

Cloud Chunk Level

In cases where only specific chunks within a Cloud Data Object have expired, Cohesity performs a more granular garbage collection process. This process varies depending on the type of external target:

Figure 13: Cloud Chunk Level



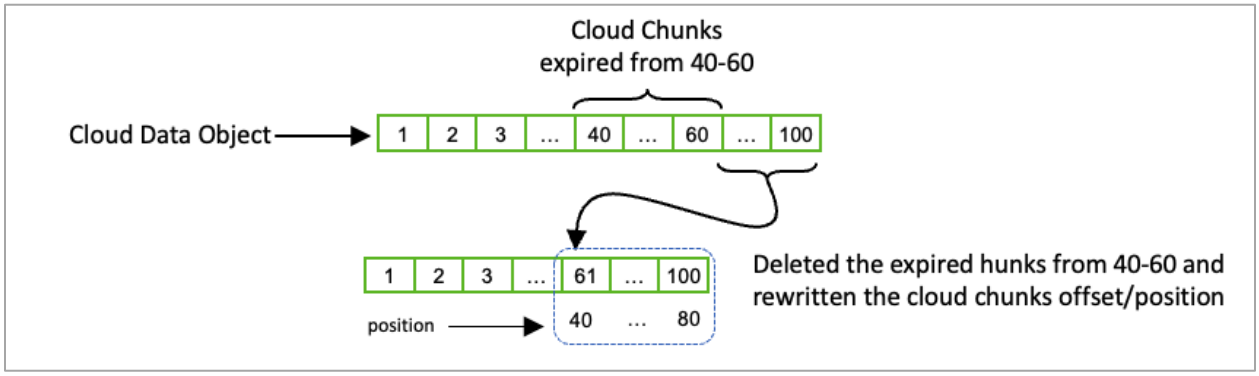
AWS S3 and Azure Blob Storage: Cohesity utilizes serverless computing services (such as AWS Lambda or Azure Functions) to delete the expired chunks and rewrite the remaining valid chunks within the Cloud Data Object.

> **NOTE**: Cohesity does not utilize AWS Lambda to manage garbage collection (GC) on colder storage tiers like S3 Glacier or S3 Glacier Deep Archive. This is because the cost of running GC operations could exceed the cost of storing data in these colder tiers.
>
> For Cohesity Cloud Edition (CE) and NextGen Cloud Edition (NGCE) clusters, Lambda or Azure Functions are used only when the external storage target is located in a different region. When the storage target is within the same region, CE and NGCE leverage an S3-compatible GC model, downloading metadata directly to the cluster for GC operations. This approach avoids Lambda/Azure Function charges, optimizing cost efficiency.
>
> If the external target is deployed in a different AWS or Azure region, Cohesity employs the respective Lambda or Azure Functions for GC to minimize inter-region data transfer and operational overhead. This intelligent, region-aware approach ensures that Cohesity optimizes costs for customers while effectively managing GC operations.

S3-Compatible External Targets: For S3-compatible storage systems, Cohesity follows a slightly different approach. The system first downloads the archive metadata from the S3-compatible target and compares it with the local cluster metadata to identify expired chunks. Once identified, the expired chunks are deleted, and the Cloud Data Object is rewritten with the valid chunks.

Here's an example of Cohesity's granular garbage collection process:

Figure 14: Granular Garbage Collection Process



Consider a Cloud Data Object that contains 100 cloud chunks, numbered 1 through 100. If chunks 40 through 60 expire and are no longer referenced, Cohesity will perform a garbage collection operation. During this process, Cohesity will rewrite Cloud Data Object, consolidating the remaining valid chunks. The new Cloud Data Object will contain chunks 1 through 39 in positions 1 through 39, followed by chunks 61 through 100, which will be shifted to positions 40 through 80.

Starting from Cohesity Version 7.2.2, Cohesity has two methods to perform Garbage Collection for S3-Compatible External Targets:

1. Storage Optimised: Increases network bandwidth as Cohesity downloads the metadata every time, to compare with the local cluster metadata for identifying expired chunks for GC calculation. Egress charges may be applicable for cloud-based S3 compatible targets.
2. Network Optimised: The Object is deleted only when all the blocks in that object have expired. This increases storage consumption; however, it reduces data transfer over the network. Minimizes bandwidth consumption.

Figure 155: Garbage Collection options in Cohesity version 7.2.2

NOTE: In scenarios where a customer's archived snapshots have expired, but the cluster still contains active local snapshots with longer retention periods, Cohesity will not perform garbage collection on the expired data from the external target. This is because the active local snapshots are referencing the archived data.

Cohesity is designed to enable faster archiving processes. By retaining the expired data in the external target, Cohesity allows for incremental archiving. Suppose a customer initiates a new archive at any point. In that case, Cohesity can efficiently reference the existing expired data, avoiding the need for a full archive and thus saving time and resources. If you want to delete the expired archive data immediately, Cohesity recommends unregistering the external target.

Metadata Management and Egress Optimization

Cohesity employs a strategic approach to metadata management to minimize egress charges associated with cloud storage:

Cloud Targets: For cloud-based archives, Cohesity stores a copy of the archive metadata within the local cluster. This reduces the need for data transfers when identifying expired archive data in the cloud, thereby minimizing egress charges. This method is particularly advantageous for cloud targets like AWS S3 and Azure Blob Storage, where egress costs can accumulate quickly.

S3-Compatible Targets: Since S3-compatible external targets are usually within the same data center, egress charges are generally not a concern. To optimize local storage, Cohesity stores the archive metadata directly on the S3-compatible target. During each garbage collection cycle, Cohesity downloads the metadata, identifies expired archive data, and rewrites Cloud Data Objects as necessary.

Recovery Considerations

To ensure a smooth data restoration operation with Cohesity, follow these best practices and guidelines:

- Cohesity recommends planning the SLA before initiating any recoveries.
- The time to retrieve data from different storage tiers varies depending on the type of storage. For instance, if data recovery is required from S3 Glacier, the first byte of data will take a minimum of four hours to be available for recovery. For S3 Glacier Deep Archive, the expected recovery time is 12 hours. For the Azure archive tier, it's 15 hours.
- During the planning phase, Cohesity suggests planning for the Recovery Time Objective (RTO) before selecting a storage class for archival purposes. If frequent recovery is expected, it's not advisable to use colder tiers.

Best Practices for Restore from Object-based Tape

- When initiating a recovery from SpectraLogic Object-based Tape, monitor alerts and emails to get the required tapes needed to complete the restore.
- Plan Object-based Cache sizing so that it can handle your restore data and ongoing backups as mentioned below:

IMPORTANT: When a Restore is initiated, the object restored from the S3-Compatible Object-based Tape is kept in the Object-based Tape Cache for 7 days by Cohesity (this is to avoid the need to rehydrate the data multiple times if the restore fails/cancels for any reason).

In Spectra Logic, the cache is managed by Spectra Logic BlackPearl and keeps the data in the Cache until midnight UTC + 1 day (i.e., if you restore at 5 pm UTC on the 20th, the data will stay in the cache until midnight UTC on the 29th).

In total, the restored objects are kept in Spectra Logic Cache for 7 + 2 = 9 days.

Please contact Cohesity Support to change the configuration setting to reduce the number of days the restored data is kept in the BlackPearl Cache.

Technical Support and Resources

- Cohesity Support Portal provides you access to a robust, on-demand, and detailed knowledge base, along with high-quality services to boost your experience with Cohesity products.
- Cohesity Product Documentation provides you with access to the latest product documentation to support your deployment of Cohesity products, including technical guides and a third-party software support matrix for Cohesity Data Protection.
- Cohesity Developer Portal provides you with ready-to-use integrations with the automation and orchestration tools you choose to streamline operations.

Your Feedback

Was this document helpful? Send us your feedback!

About the Authors

Saran Ravi is a Sr Solutions Architect at Cohesity. In his role, Saran focuses on Cloud and Kubernetes.

Jedidiah Sonavane is a Solutions Architect at Cohesity. In his role, he focuses on S3-OnPrem, Multitenancy/Service Provider, and Cloud Archival.

Other essential contributors include:

- Adaikkappan Arumugam, Sr Director Product Solutions
- Anirudh Kumar, Technical Director
- Dayanand Sharma, Director Product Management
- David Jayanathan, Field Technical Director
- James White, Principal Solutions Engineer
- Kevin Hill, Manager, Solution Architects
- Shayne Wiliams, Principal Architect
- Karan Naik, Senior Site Reliability Engineer
- Bharath Nagraj, Senior Principal Field Technical Director
- Edwin Galang, Solution Architect
- Anupam Sharma, Staff 2 Engineer, QA

Document Version History

VERSION	DATE	DOCUMENT HISTORY
2.0	Aug 2025	Content update – S3-Compatible and Object-tape
1.6	Jan 2025	Updated AWS Lambda details.
1.5	Sep 2024	Added the “Garbage Collection Considerations” section.

VERSION	DATE	DOCUMENT HISTORY
1.4	Aug 2024	Content update
1.3	July 2024	Republishing
1.2	Nov 2023	Content update
1.1	Apr 2023	Content update
1.0	Jan 2023	First full release

ABOUT COHESITY

Cohesity is a leader in AI-powered data security and management. Aided by an extensive ecosystem of partners, Cohesity makes it easier to protect, manage, and get value from data – across the data center, edge, and cloud. Cohesity helps organizations defend against cybersecurity threats with comprehensive data security and management capabilities, including immutable backup snapshots, AI-based threat detection, monitoring for malicious behavior, and rapid recovery at scale. Cohesity solutions are delivered as a service, self-managed, or provided by a Cohesity-powered partner. Cohesity is headquartered in San Jose, CA, and is trusted by the world's largest enterprises, including six of the Fortune 10 and 44 of the Fortune 100.

Visit our website and blog, follow us on Twitter and LinkedIn, and like us on Facebook.

© 2025 Cohesity, Inc. All rights reserved.
Cohesity, the Cohesity logo, SnapTree, SpanFS, DataPlatform, DataProtect, Helios, the Helios logo, DataGovern, SiteContinuity, DataHawk, and other Cohesity marks are trademarks or registered trademarks of Cohesity, Inc. in the US and/or internationally. Other company and product names may be trademarks of the respective companies with which they are associated. This material (a) is intended to provide you information about Cohesity and our business and products; (b) was believed to be true and accurate at the time it was written, but is subject to change without notice; and (c) is provided on an "AS IS" basis. Cohesity disclaims all express or implied conditions, representations, warranties of any kind.