NexaVM Technologies AG.

NEXAVM

nSDS v5.4.0

Software Defined Storage
Version: 5.4.0

Technical White Paper

January 2026

Table of Contents

Copyright Statement 2
1 Product Overview 5
2 Product Architecture 6
2.1 System Architecture 6
2.1.1 Hardware Layer 7
2.1.2 Platform Layer 7
2.1.3 Data Layer 8
2.1.4 Interface Layer 9
2.1.5 Business Layer 9
2.2 Key Components 9
2.2.1 Distributed Scheduling Component 10
2.2.2 nSDS-master 10
2.2.3 nSDS-UI 10
2.2.4 zstnlet 10
2.2.5 zstnctl 11
2.2.6 zstnadm 11
2.2.7 nSDS-alertmanager 11
2.2.8 nSDS-pushgateway 11
3 Features 12
3.1 Data Storage 12
3.1.1 Caching Acceleration Scheme 12
3.1.2 Thin Provisioning Automation 12
3.1.3 Set Quality of Service (QoS) for Storage Pool Recovery 13
3.1.4 Volume Business QoS 13
3.2 Data Protection 14
3.2.1 Volume Snapshot 14
3.2.2 Data Duplication 15
3.2.3 Failure Domain Isolation 24
3.2.4 Cluster Hardware Topology 26
3.2.5 Data Consistency Inspection 27
3.3 Convenient O&M 27
3.3.1 Multiple Resource Pools 27
3.3.2 Hard Disk Beacon 27
3.3.3 Disk S.M.A.R.T. Detection 28
3.3.4 Data Rebalancing 28
3.3.5 Data Volume Maintenance Mode 28
3.3.6 Automatic Fault Detection and Alarm 28
3.4 Product Integration 29
3.4.1 NexaVM Cube Integration 29
3.4.2 NexaVM Edge Integration 30
3.4.3 Integrating a Third-Party Storage Platform 30

3.4.4 Virtualization and Cloud Native Platform Integration 30
4 File Storage Service 31
4.1 File System 31
4.2 File Sharing 33
5 Block Storage Service 34
5.1 Third-Party LUN Take-Over 34
5.2 Volume Migration 34
5.3 Volume Clone 35
6 Object Storage Service 35
6.1 S3 Interface 35
6.2 Access Control Management 36
6.3 Multipart Upload 38
6.4 Storage Policy 38
6.5 Data Compression 38
7 Feature 40
Glossary 46

1 Product Overview

NexaVM nSDS is a distributed storage software for modern infrastructures. You can deploy it on any conventional server to form a reliable storage system that meets your application performance and interface requirements. NexaVM nSDS provides rich resource management features, including management of servers, hard disks, data disks, block storage volumes, storage pools, and storage buckets. You can use these features to create storage pools with different types of data redundancy and storage buckets with specified storage policies and access permissions, thus enabling you to quickly and cost-effectively build a simple, stable, and secure storage infrastructure.

NexaVM nSDS, as a product of current information technology development, brings new possibilities for data storage and processing in scenarios such as virtualization, cloud platforms, cloud-native, and big data. Its distinctive advantages are mainly reflected in the following aspects:

- **Elastic Scalability**:

Users can easily scale storage resources horizontally. As business grows, data expansion is often inevitable. Traditional centralized storage systems face significant challenges when dealing with large-scale data growth, while distributed storage systems can simply add nodes to increase the capacity and processing power of the entire storage cluster. This elastic scalability enables NexaVM nSDS to adapt to business needs of various scales.

- **High Availability**:

Supports multiple data redundancy strategies, including replication and EC (Erasure Coding). The replication strategy supports online adjustment of storage pool replica counts, copying data across multiple nodes. This ensures data remains accessible from other nodes when one or several nodes fail, maintaining cluster availability. Combined with flexible failure domain strategies, NexaVM nSDS can guarantee continuous storage service even during hardware failures or network outages.

- **Performance Optimization**:

Utilizes proprietary cache acceleration technology to optimize data processing performance. Enterprise-grade NVMe SSDs and SATA SSDs accelerate read/write operations for backend HDD devices, effectively reducing latency and improving cluster I/O performance. Additionally, NexaVM nSDS employs load balancing to distribute storage I/O evenly across nodes, ensuring optimal resource utilization.

- **Easy Management**:

Provides user-friendly management interfaces and standard RESTful API interfaces, enabling administrators to easily perform basic lifecycle management of physical resources, as well as data operations and recovery tasks.

- **Cost-Effective:**

Supports deployment on commodity hardware, significantly reducing costs. Its elastic scalability allows dynamic resource adjustment based on business needs, preventing resource waste and further lowering operational expenses.

In summary, NexaVM nSDS delivers an efficient, scalable, reliable, and cost-effective solution for enterprise data storage and processing. Whether for big data processing, cloud computing, AI, or other emerging technologies, NexaVM nSDS distributed storage software is an ideal choice.

2 Product Architecture

2.1 System Architecture

NexaVM nSDS delivers industry-leading performance, reliability, and manageability that powers large enterprises and service providers to build storage pools with petabyte-plus capacity.

NexaVM nSDS architecture has the following advantages:

- **Hybrid Cloud Native**:

  The block storage service provides RBD and iSCSI interfaces to connect to various virtualization and native cloud platforms and serve as a solid cloud foundation. The object storage service has native cloud capabilities that allow you to deploy it in private, public, hybrid, and multi-cloud environments and at the edge with a seamless and consistent experience.

- **Comprehensive Monitoring**:

  An independent monitoring system is provided to monitor storage from hardware to network in real time, helping you keep track of storage performance and ensure data security.

- **Low Hardware Dependency**:

  Software-defined storage pools various types of x86 servers, including old servers and homegrown servers, reducing vendor lock-in.

- **High Performance**:

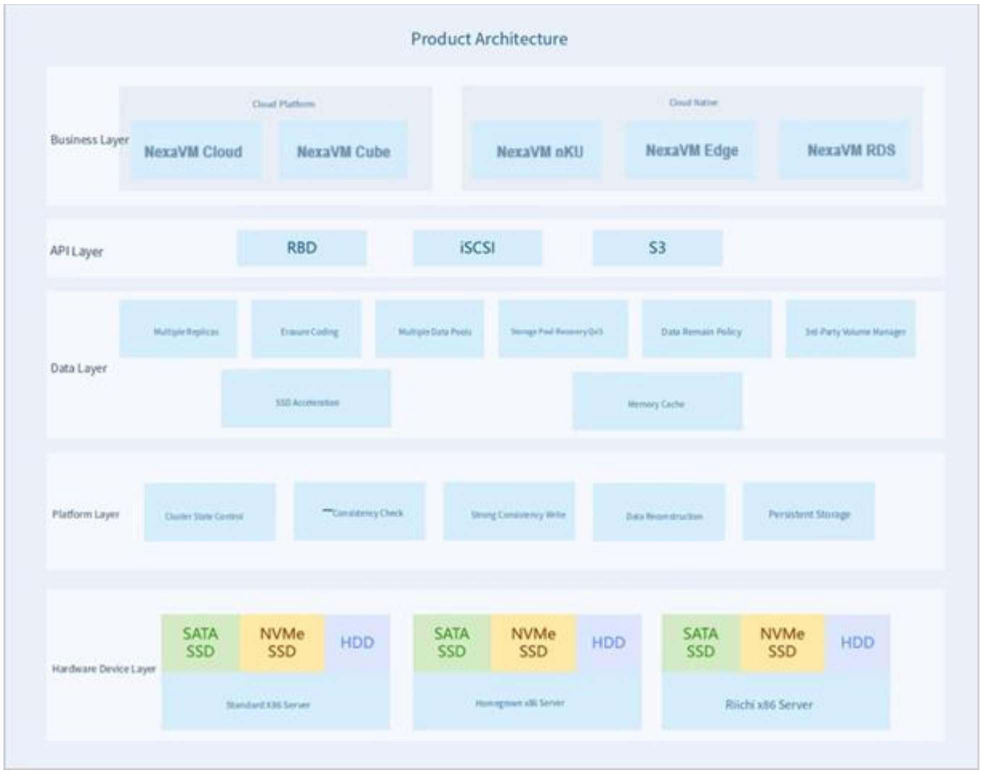
  An SSD cache is used to accelerate backend HDDs by leveraging their low latency and high IOPS. Performance and capacity scale linearly with server scale.

- **High Reliability**:

  Data is not lost when a server crashes because of data redundancy. Data consistency is guaranteed by strong write.

NexaVM nSDS logical architecture is shown in *Figure 2-1: System Architecture*.

Figure 2-1: System Architecture



2.1.1 Hardware Layer

NexaVM nSDS distributed storage software supports x86 as the underlying hardware platform. At the same time, storage allows for legacy pooling to protect your existing IT assets.

- Hard Disk: The storage software supports mainstream enterprise-level SATA SSD, NVMe SSD, and SATA HDD.
- Network: The storage software supports mainstream NICs and switches such as 1 Gigabit, 10 Gigabit, and 25 Gigabit.

2.1.2 Platform Layer

NexaVM nSDS uses the consistency hashing algorithm with fault domains to solve the consistency hashing problem in storage services. To be more specific, a fault domain-based algorithm adds physical deployment logics and cluster state parameters on the original hash algorithm, thereby minimizing data migration when cluster data changes. For example, based on the physical and logical topology that you enter, the fault domain-based algorithm can selectively migrate data within a server or within a chassis to minimize the resource consumption caused by migration. In addition, data is distributed based on the cluster state, which can make up for the differences of various storage devices.

2.1.3 Data Layer

NexaVM nSDS applies self-developed caching technologies to read and write data to enterprise SATA SSD or NVMe SSD for backend HDDs. Hot data is stored in the SSDs while cold data is smartly written to HDDs based on data access frequency. This not only improves the storage cluster performance but also provides vast storage space for upper-layer applications.

- **Write data:**

For write operations, data is first written to the write cache on the SSD. After all replicas in the write cache are successfully written, the I/O responses are immediately returned. The data cache is evenly distributed across the data volumes of each server. In addition, the data volumes will periodically write the cached write I/O data on the SSD cache to the HDDs in bulk. If the data in the cache exceeds the preset watermark, the data will also be written to the HDDs.

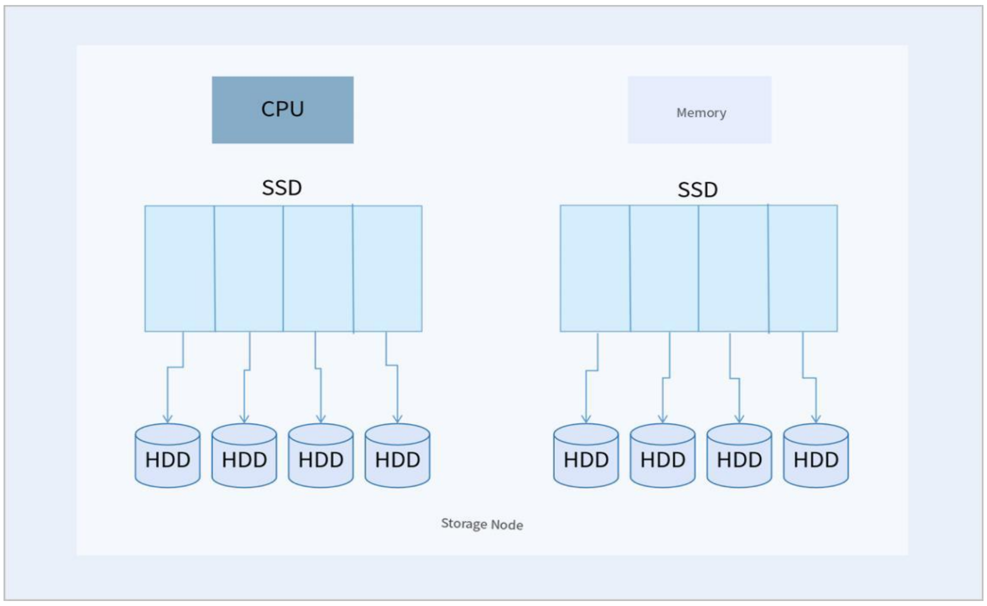
- **Read data:**

For read operations, NexaVM nSDS applies a multi-level caching mechanism consisting of RAM cache and SSD cache to lower the access frequency to HDDs and improve storage cluster read performance.

- **Data write-to-disk policy:**

Write-to-disk is the process of writing data in the SSD cache to the HDDs. Until the data is written to the HDDs, it still can be read from the SSD cache. However, it will be reclaimed if the cache policy identifies it as infrequently accessed cold data.

Figure 2-2: Storage Node



Around self-developed caching technologies, NexaVM nSDS provides a variety of management features and maintenance functions:

Supports multiple storage pool management on the same platform.

- This allows you to customize data redundancy types for storage pools, including replica and erasure coding.
- Allows you to customize topologies and multiple fault domains, including server, rack, and data center.
- Allows you to customize storage pool recovery QoS policies when a storage pool is recovering.
- Supports unified management of native storage volumes and 3rd-party storage volumes.

2.1.4 Interface Layer

The interface layer of NexaVM nSDS provides two types of interfaces:

- Block Storage: Through the RBD interface, the interface layer provides virtual volume devices to operating systems, cloud platforms, and databases.
- Object Storage: Through the S3 interface, the interface layer provides S3-based unstructured data services to third-party applications.

2.1.5 Business Layer

NexaVM nSDS connects to upper-layer businesses through block storage interfaces and object storage interfaces.

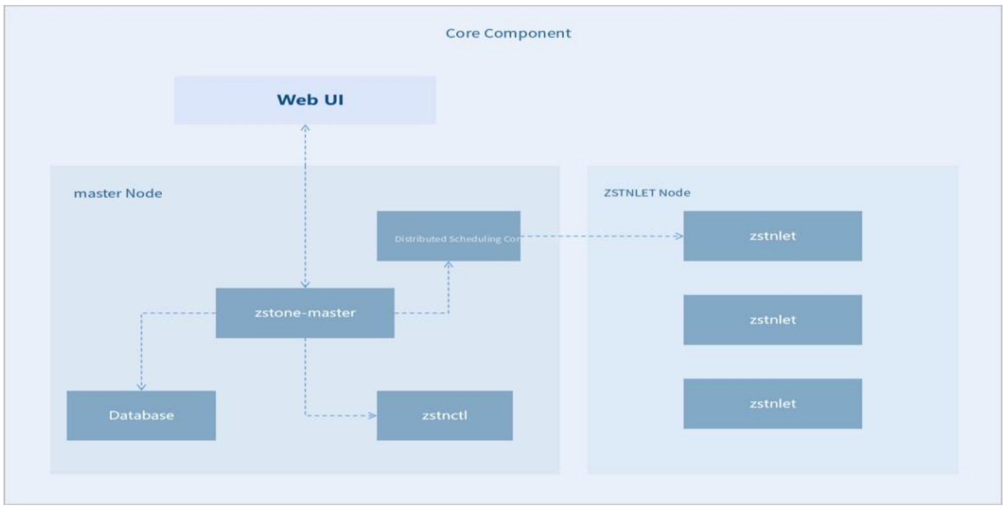
Classic business scenarios include:

- Virtualized platform: NexaVM Cloud, NexaVM Cube, and others can be connected.
- Cloud Native Platform: NexaVM nKU, NexaVM Edge, NexaVM RDS, and others can be connected through the CSI plugin.
- 3rd-Party Application: This layer connects application scenarios with non-structured data such as video surveillance, medical images, and backup archiving. S3 interfaces are used for connections.

2.2 Key Components

Distributed storage systems follow a master-slave architecture and are composed of multiple Masters and Nodes. The entire NexaVM nSDS can be divided into a control plane and numerous managed Nodes.

Figure 2-3: Key Components



2.2.1 Distributed Scheduling Component

The distributed scheduling component is a commonly used distributed system component that serves as a task engine. It leverages the persistence and communication capabilities provided by NexaVM nSDS to achieve workflow orchestration, client/server architecture, and state management. The distributed scheduling component is used to uniformly schedule business tasks in NexaVM nSDS systems, orchestrating workflows, and scheduling. It also realizes the collection and distribution of distributed tasks.

2.2.2 nSDS-master

The nSDS-master component is one of the key core components of NexaVM nSDS, providing all platform API capabilities. Installed on the management node, it offers RESTful API interfaces for platform management, authentication and authorization, license services, admission control, and security checks. It also manages the business capabilities of NexaVM nSDS clusters.

The NSDS-master component has the following features:

- Supports HA mode.
- It is decoupled from the data plane and supports separate deployment from the data plane.
- Provides standard RESTful API interfaces based on Swagger.

2.2.3 nSDS-UI

The nSDS-UI component is based on the React.JS framework, deploys on the NGINX server, and provides a Web interface for users to access and log in. By calling the RESTful API interface, it realizes operations on resources in the NexaVM nSDS platform from the Web interface. Thanks to the micro-frontend architecture, it can more easily integrate with other virtualization platforms and cloud-native platforms, achieving seamless integration of the management pages of NexaVM nSDS.

2.2.4 zstnlet

The zstnlet component is installed on the compute node and is mainly responsible for communications with the management node and execution of the scheduled tasks on the compute node. The zstnlet component uses the gRPC-based communication protocol, which improves efficiency and stability while avoiding security risks and vulnerabilities.

The zstnlet component is responsible for scheduling and managing various computing resources , including servers, hard disks, data disks, storage pools, and data volumes. It also manages the lifecycle of other services running on the compute node, such as starting/stopping servers and lifting data nodes.

2.2.5 zstnctl

The zstnctl component is an independent management tool in NexaVM nSDS, primarily used for underlying maintenance work by operations and maintenance personnel.

The zstnctl component mainly includes three types of management tools:

Storage cluster management tool, supporting the management of storage cluster services and configurations, including:

- Configuration management, enabling batch of configurations in the cluster.
- Monitoring node service management, enabling batch startup and shutdown of monitoring services.
- Management node service management, enabling batch startup and shutdown of management services.
- Object storage gateway service management, enabling batch startup and shutdown of object storage gateway services.
- Cluster cleanup tool, one-click cleanup of the current cluster.
- Object storage cleanup tool, supporting one-click cleanup of all object storage-related resources, including object users, storage buckets, and object storage system pools.
- Cluster cache configuration tool, supporting batch modification of cache configurations in the cluster.

2.2.6 zstnadm

The zstnadm component is the NexaVM nSDS service management tool, primarily responsible for the cross-version upgrading feature and the intra-version upgrading feature of NexaVM nSDS services.

2.2.7 nSDS-alertmanager

The nSDS-alertmanager component mainly provides alarm messages for NexaVM nSDS. This component is installed on the management node and links with Prometheus. When the component discovers that monitoring indicators are abnormal and meet the alarm rules, it will generate alarm messages and push alarm messages to the related API interface of NexaVM nSDS.

2.2.8 nSDS-pushgateway

The nSDS-pushgateway component is installed on the management node and is mainly responsible for aggregating monitoring metrics of all servers on NexaVM nSDS, which are then provided to Promethues. Promethues obtains the monitoring data by accessing the HTTP interface provided by the nSDS-pushgateway component.

3 Features

3.1 Data Storage

3.1.1 Caching Acceleration Scheme

A caching acceleration scheme generally refers to the use of SSD-based caching devices to accelerate traditional HDD devices. NexaVM nSDS self-developed caching acceleration technology (NAS caching) uses high-speed SSD devices to cache I/O data for traditional HDD devices, caching frequently accessed hot data to the high-speed SSD devices to improve I/O performance, especially in scenarios with hot data access patterns.

The NAS caching module supports two caching strategies: write-through and write-back. In the write-through mode, data is written to both the cache and the backend storage device to ensure data consistency, while in the write-back mode, most of the cache is used to buffer written data and ensure that dirty data is written in order to the backend storage device. By default, the write-back strategy is disabled, and you can switch the caching strategy at runtime.

The NAS caching module has the following features:

- **Smart Data Migration**

  The NAS caching module automatically migrates databased on its access frequency and to SSD caches to improve access speed. At the same time, the caching module automatically migrates less frequently used data from the SSD cache to the slow disk based on the cache usage and space constraints to free up cache space.

- **Data Protection**

  When data I/O errors occur on the flash, the NAS caching module will first attempt to read data from the disk to recover the data or mark the cache item as invalid. For irrecoverable errors, such as metadata or dirty data, the caching module will automatically disable the cache.

3.1.2 Thin Provisioning Automation

NexaVM nSDS provides thin provisioning automation that virtualizes a greater amount of storage than what is physically available on the actual storage device and writes data to the logical volume. This feature makes storage more convenient to scale and effectively saves storage space.

3.1.3 Set Quality of Service (QoS) for Storage Pool Recovery

Quality of Service (QoS) is a technology used to solve I/O resource allocation issues. With QoS, you can set different bandwidth limits for different types of application workloads to achievebetter resource allocation.

In the Data Node, an op_shardedwq queue processes all kinds of upper-layer I/Os. This queue is a composite queue that usually includes several sub-queues. After an I/O request is dequeued from the queue, it is implemented with the ObjectStore interface and the disk. Two major types of I/O requests are involved, one from the client and the other generated from internal system activities, including inter-DataNode I/O requests, SnapTrim, Scrub, and Recovery.

NexaVM nSDS uses a weighted priority queue (WPQ) to classify the I/Orequests into different sub-queues according to the preceding classification. When a request isenqueued into a WPQ for the first time, the WPQ is created. When a WPQ is dequeued, thepriorities of the WPQs are used as weights to determine the WPQ to bedequeued. However, even if a WPQ is selected, it does not necessarily dequeue a request, for the size of the request to bedequeued also needs to be considered.

You can set a QoS for storage pool recovery. Three QoS levels are available:

- Slow: This QoS ensures business bandwidth first and the recovery time is long. In case
- of hardware failures during the recovery, the data safety level may be lowered. We recommend you use this QoS in production environments.
- Medium: This QoS ensures both business and recovery bandwidths. The recovery time is medium. If the system performance is saturated, the I/O latency may be increased.
- Fast: This QoS ensures recovery bandwidth first and the recovery time is short. If the system performance is saturated, the business performance may be affected.

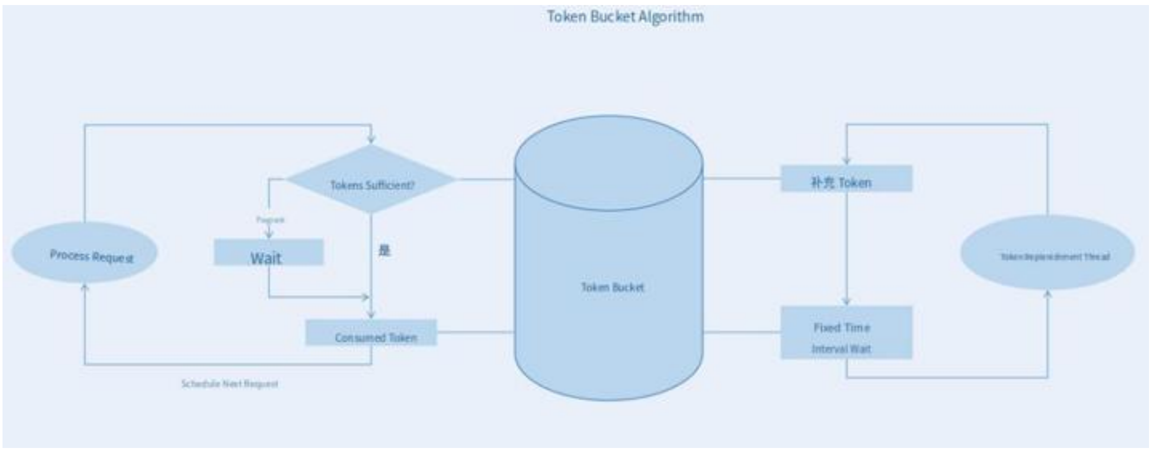
3.1.4 Volume Business QoS

You can set business QoS for a block storage volume, including maximum IOPS and maximum bandwidth. By setting a business QoS, you can govern the performance of different volumes and satisfy diverse business performance requirements.

NexaVM nSDS uses the token bucket algorithm to govern the I/O traffic:

- The system generates tokens at a fixed rate and puts the tokens into the bucket. Every I/O request must obtain a token from the bucket.
- I/O requests that fail to obtain tokens are put into a queue to wait for tokens. This mechanism governs the average data flow into the system.
- If the rate at which the system distributes tokens is lower than the rate at which the system generates tokens, the bucket may accumulate some tokens. When a sudden data flow occurs, the system allows you to directly use the tokens in the bucket.

Figure 3-1: Token Bucket Algorithm



3.2 Data Protection

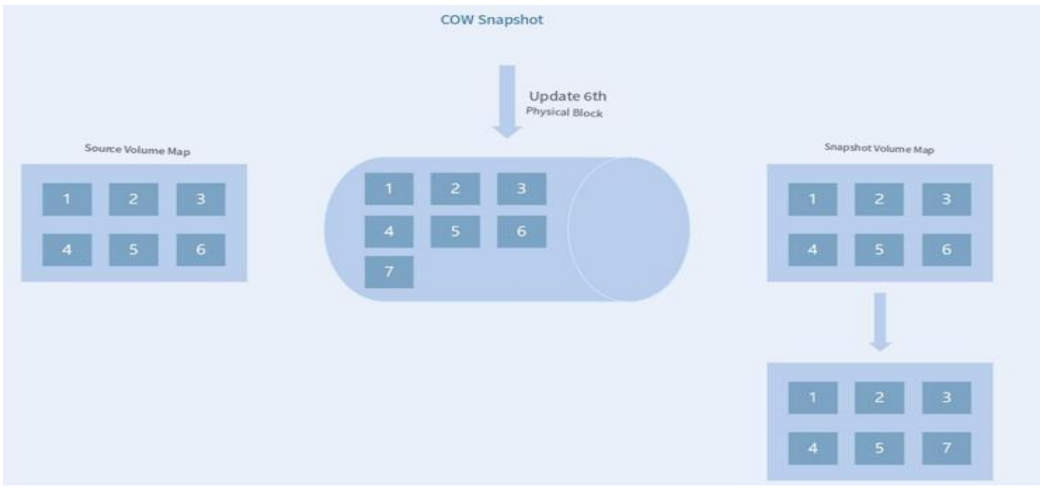
3.2.1 Volume Snapshot

A snapshot is a key feature of NexaVM nSDS, which allows you to create a read-only replica of a block storage volume at a specific time point without interrupting services or affecting the production environment.

Snapshot Creation

NexaVM nSDS uses COW (Copy-On-Write) snapshot technology. After a COW snapshot is created, the system creates a read-only replica of the original block device in the storage cluster. Only when the data in the original block device or the snapshot is changed, will the system copy the changed data blocks. This way, the original data is reserved in the snapshot and redundancy is avoided. At the same time, the system records the relationship between the original block device and the snapshot, as well as metadata information of each snapshot, including the creation time, size, and parent snapshot. These metadata records help the system locate the specific snapshot state when restoring data.

Figure 3-2: COW Snapshot



Data Rollback and Recovery

You can create new block storage volumes based on a snapshot, which are clones. In addition, in case of a disaster or data rollback, you can also roll back a volume to the state at a specific snapshot point.

3.2.2 Data Duplication

Overview

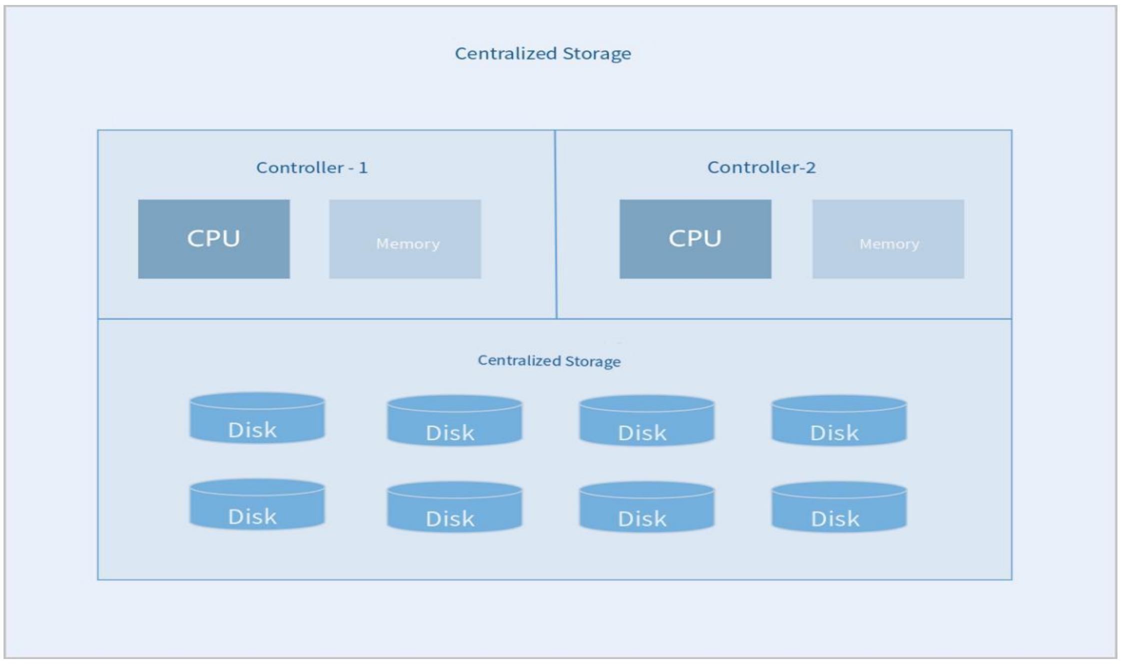
Data redundancy is the cornerstone of achieving high availability, data protection, and business continuity in storage systems. As data volumes grow, application scenarios become more complex, and potential threats such as hardware failures, network interruptions, and human errors increase, an appropriate data redundancy strategy can serve as a robust data protection barrier, ensuring business continuity and maximizing data value. This chapter briefly compares the redundancy features of centralized and distributed storage and then elaborate show NexaVM nSDS employs two core data security policies—redundant copies and erasure coding (EC).

Comparison between Distributed Storage and Centralized Storage

Centralized Storage

Traditional centralized storage uses a controller and disk cabinet to provide data management and read/write capabilities. Generally, dual controllers are used for redundancy, with some high-end storage systems featuring multiple controllers. Storage space can be provided through controller- native disk slots or by connecting an external disk cabinet. Traditional centralized storage typically employs RAID technology such as RAID 5, RAID 6, and RAID 10 to protect data.

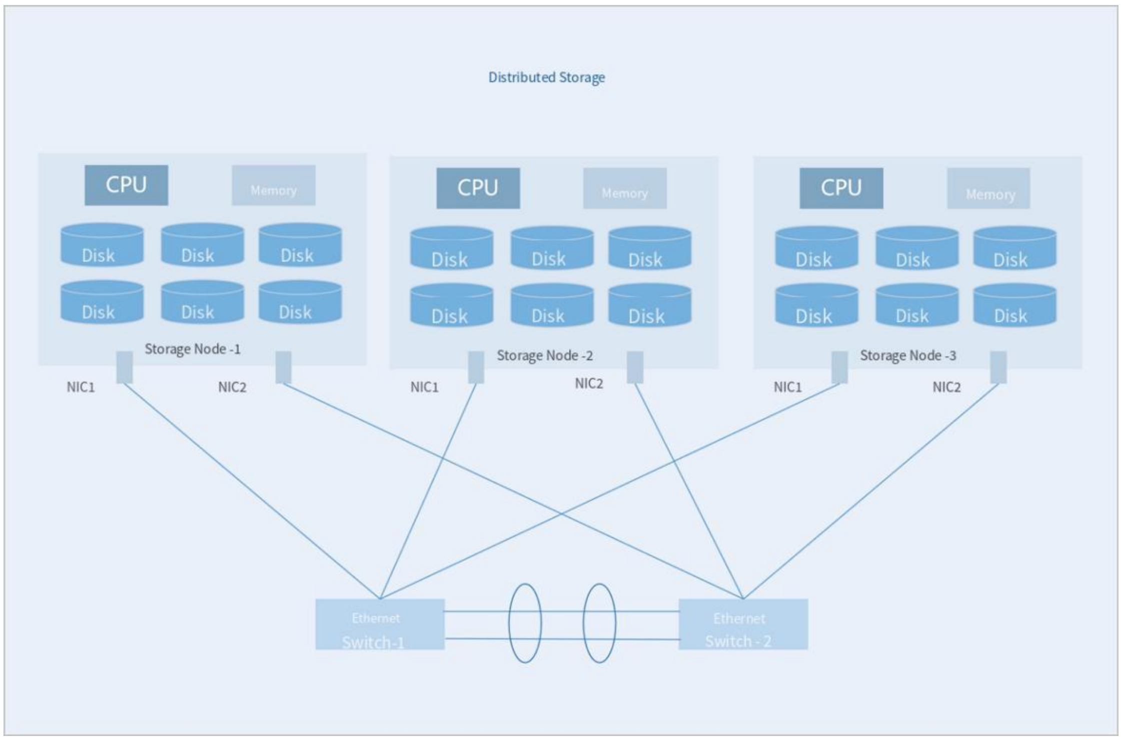
Figure 3-3: Centralized Storage



Distributed Storage

Distributed storage adopts a non-centralized networking method, with each storage node capable of providing both computing and storage resources for flexible scalability and a larger storage scale. Storage nodes are interconnected via generic Ethernet switches and provide a unified storage resource pool to upper-layer businesses based on distributed storage software. In addition, distributed storage supports horizontal scaling, with a single cluster potentially expanding to thousands of nodes to provide EB-level capacity, making it suitable for storing massive amounts of data.

Figure 3-4: Distributed Storage



Centralized Storage vs. Distributed Storage

- **Node-to-Node Replication**: Distributed storage supports node-to-node replication, such as 3 replicas that can tolerate the failure of two nodes without losing data, whereas RAID is limited to disk redundancy within a single node.
- **Global Hot Standby and Data Recovery**: Unlike RAID, distributed storage does not require dedicated hot standby disks, as all disks participate in data recovery, offering significantly higher efficiency. Moreover, distributed storage does not require additional hardware support, unlike RAID, which needs independent RAID cards.

Volume Replica

Definition

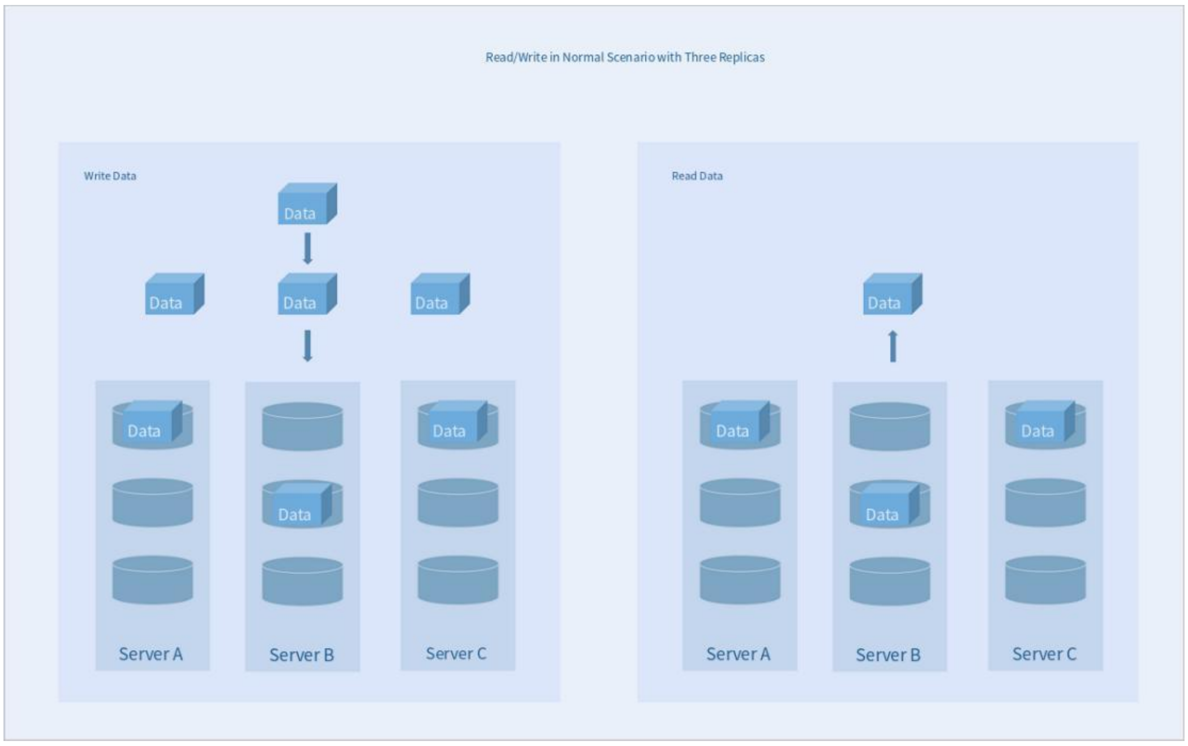
A data protection technique that creates identical data on different nodes to achieve data redundancy and high availability. When a node fails, data can be recovered from the replica on another node. NexaVM nSDS allows you to set 2–6 replicas for your volumes. We recommend that you set 3 replicas for your production environment.

Read/Write Principle

- **Normal Read/Write**

  With a 3-replica strategy, when data is written, the system writes 3 identical replicas to three data volumes on 3 servers. When data is read, the system reads the data from one of the 3 servers and returns the data to the primary storage.

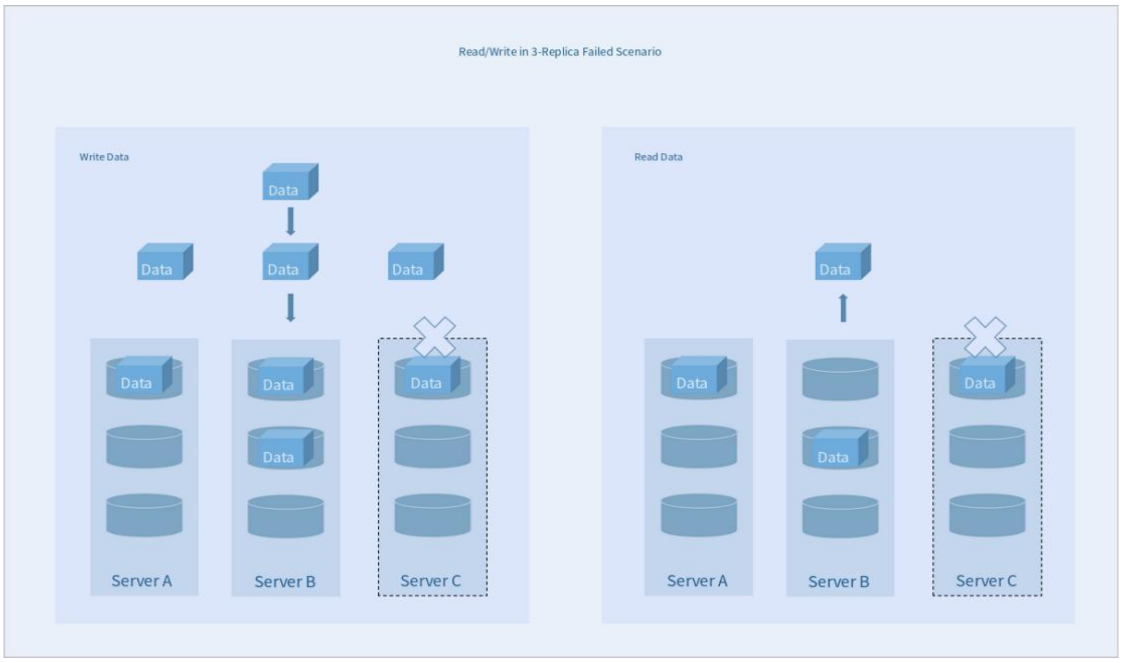
Figure 3-5: Normal Read/Write with 3 Replicas



- **Failed Read/Write**

  With a 3-replica strategy, when a server fails, the system writes the replicas to the remaining 2 servers. When data is read, the system reads the data from one of the 2 servers and returns the data to the primary storage.

Figure 3-6: Failed Read/Write with 3 Replicas



Erasure Coding

Overview

Erasure coding (EC) is a data protection technology that divides data into K data chunks and generates M parity chunks by using a fault-tolerant algorithm. This approach enables data recovery from failures. Compared with the traditional replica-based approach, EC achieves data reliability while saving storage space and network bandwidth.

- Standard EC (K+M): K represents the number of data chunks, and M represents the number of parity chunks. This EC strategy indicates that the data can work as expected when up to M fault domains fail.
- Folded EC (K+M:B): K represents the number of data chunks, M:B represents the number of parity chunks. This EC strategy indicates that the data can work as expected when up to M disks or B fault domains fail.

EC Policy

NexaVM nSDS provides the following EC policies:

EC Policy		Disk Saving Rate
Recommended	2+1	66.67%
	4+2	66.67%
	8+3	72.73%
	4+2:1	66.67%
	8+2:1	80.00%
	16+2:1	88.89%
Custom		K/(K+M)

Standard EC

Read/Write Principle

- Normal Scenario

Standard EC (K+M): Using a 4+2 EC policy as an example, when writing data, the system divides the data into 4 data chunks of the same size and generates 2 parity chunks of the same size using the EC algorithm. Then, the system randomly stores these 6 chunks on 6 servers. If any 2 servers fail, the data can still be used normally. When reading data, the system reads data blocks from different disks of 4 running servers, assembles the 4 data chunks into complete data, and returns the data to the user.

Figure 3-7: Standard EC (4+2) Normal Scenario Write Data

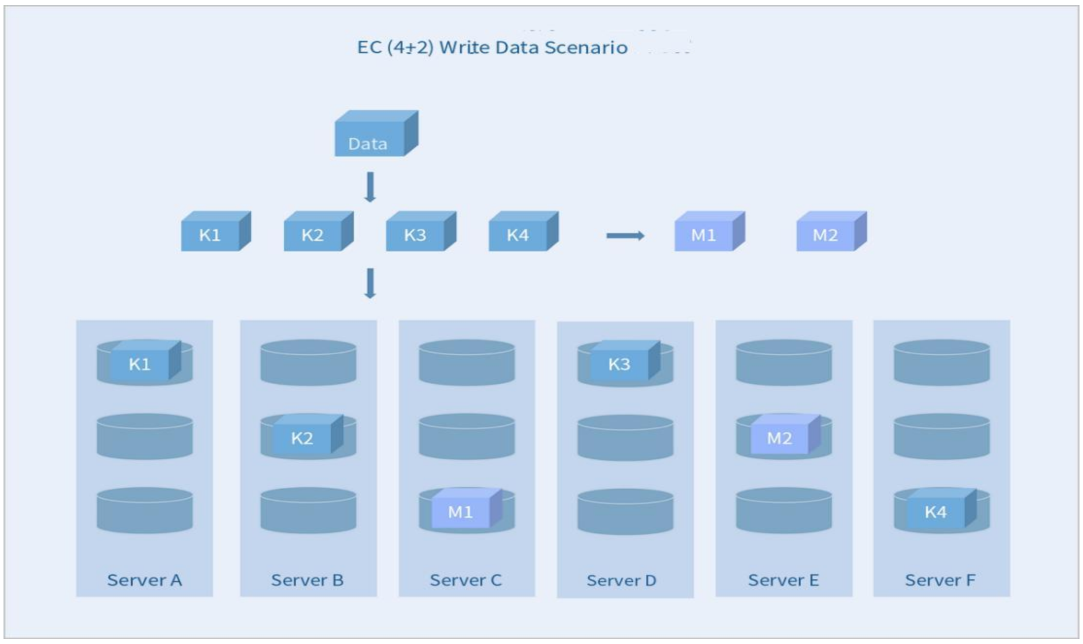
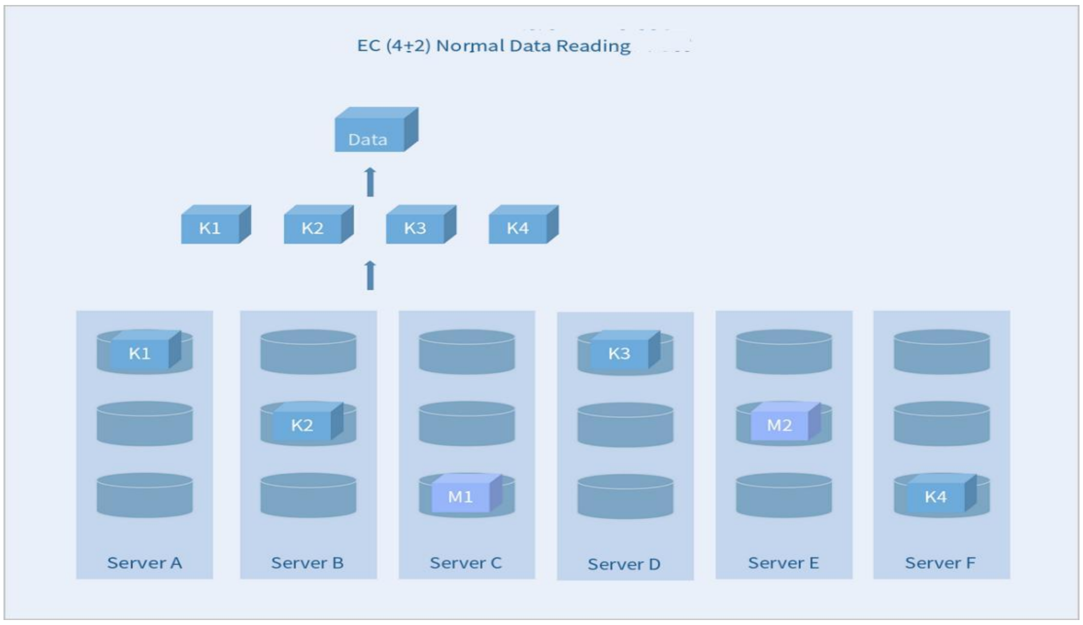


Figure 3-8: Standard EC (4+2) Normal Scenario Read Data



- Failure Scenario

Standard EC (K+M): Using a 4+2 EC policy as an example, when the number of remaining servers is less than K+M after a failure, the system stores newly written data on the remaining servers before the recovery to ensure that I/O is not interrupted and the data reliability level is not reduced. After the failure is recovered, the data redundancy policy returns to K+M. When reading data, the system reads data from other normal servers and recovers the data using the EC algorithm before returning it to the user.

Figure 3-9: Standard EC (4+2) Failure Scenario Write Data

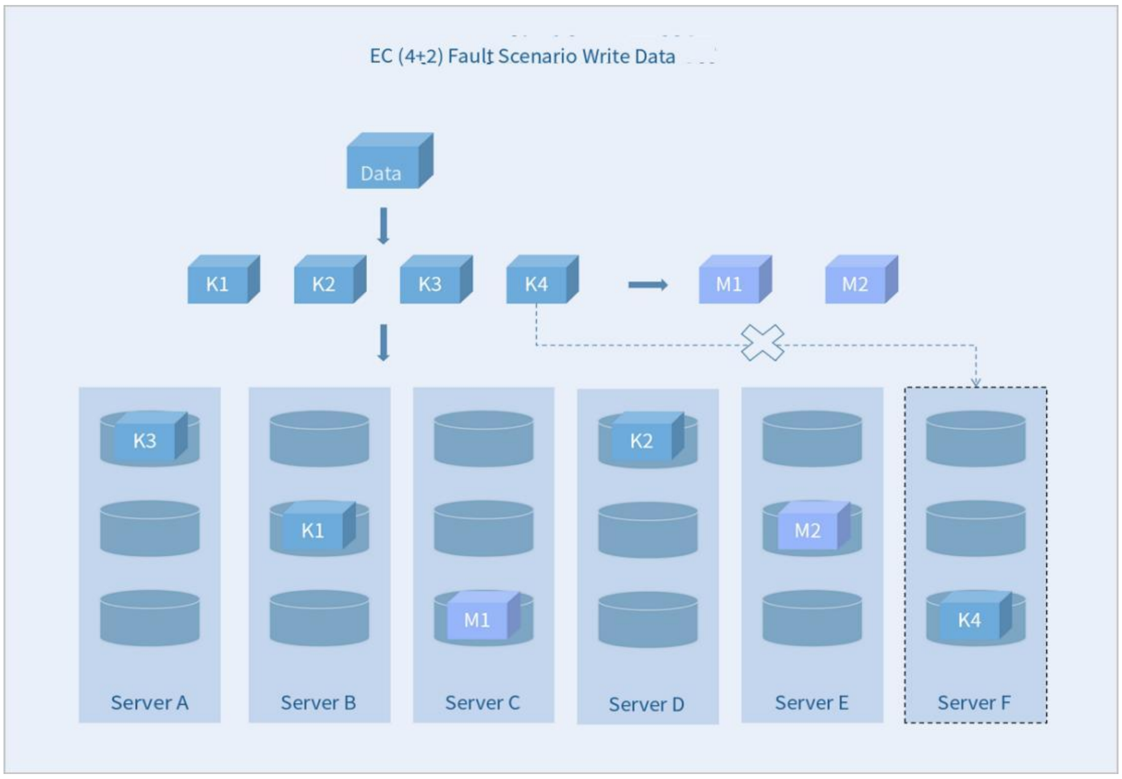
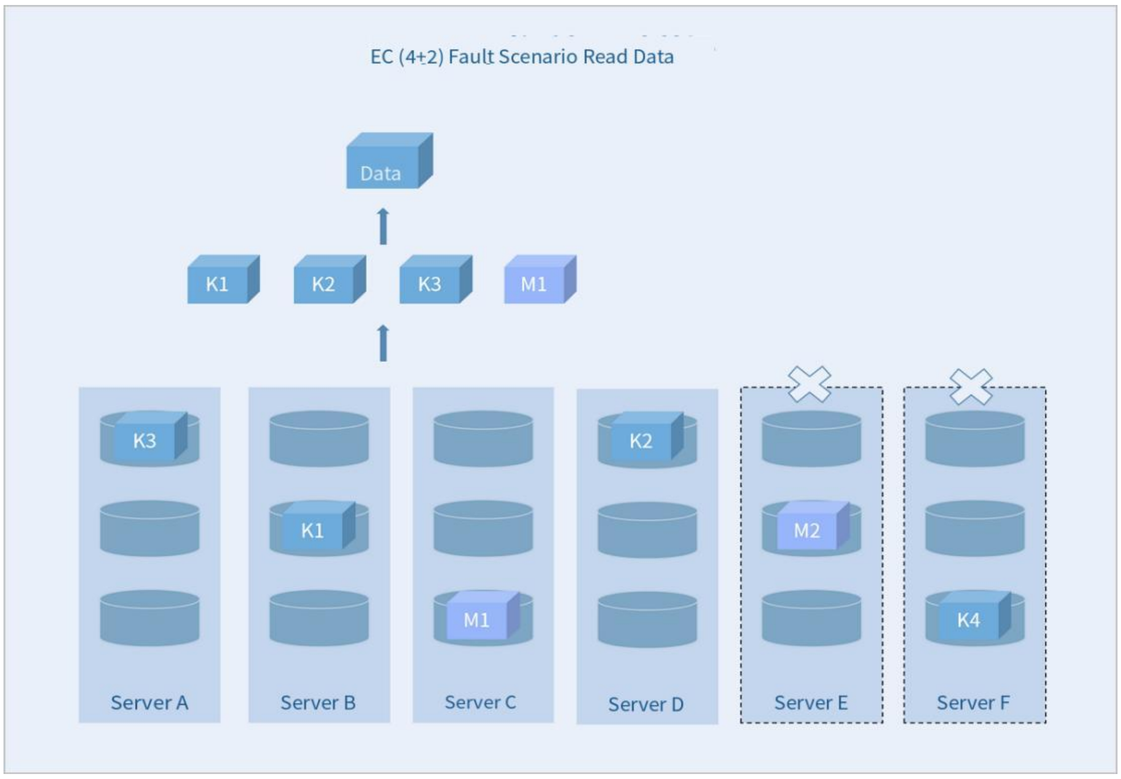


Figure 3-10: Standard EC (4+2) Failure Scenario Read Data



Stacked EC

Stacked EC, also known as sub-node EC, is a commonly used data redundancy technology. Unlike standard EC with a K+M ratio, stacked EC typically uses a K+M:B ratio, where B is usually 1. Similar to standard EC, stacked EC ensures high data reliability while maintaining high disk utilization.

For example, a standard EC has a minimum fault domain of a storage node, so the minimum mode also requires at least 6 nodes. In contrast, the 4+2:1 ratio of stacked EC only needs 3 storage nodes to meet the data redundancy requirements.

In addition, NexaVM nSDS allows you to scale in or out the capacity of stacked EC and convert it to standard EC if it meets the fault domain requirement.

Write and Read Principle

- Normal read/write

Stacked EC (K+M:B): Using the server-level 4+2:1 EC strategy as an example, when writing data, the system divides the data into 4 data fragments of the same size and generates 2 check fragments of the same size through the check algorithm. Then, the system randomly stores these 6 data fragments into 5 servers. When any 1 server fails, the data can still be used normally. When reading data, the system reads data blocks from different disks of 3 servers and assembles these data blocks into complete data before returning them to the user.

Figure 3-11: Stacked EC (4+2:1) Normal Write

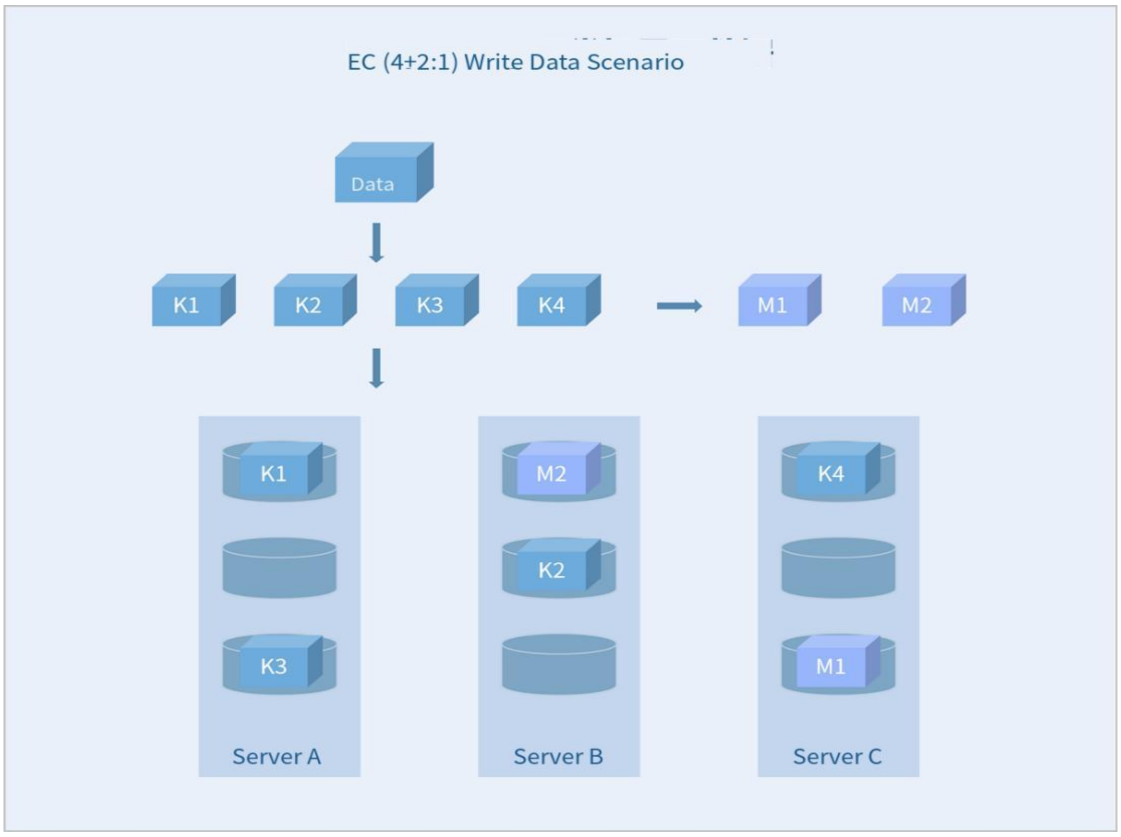
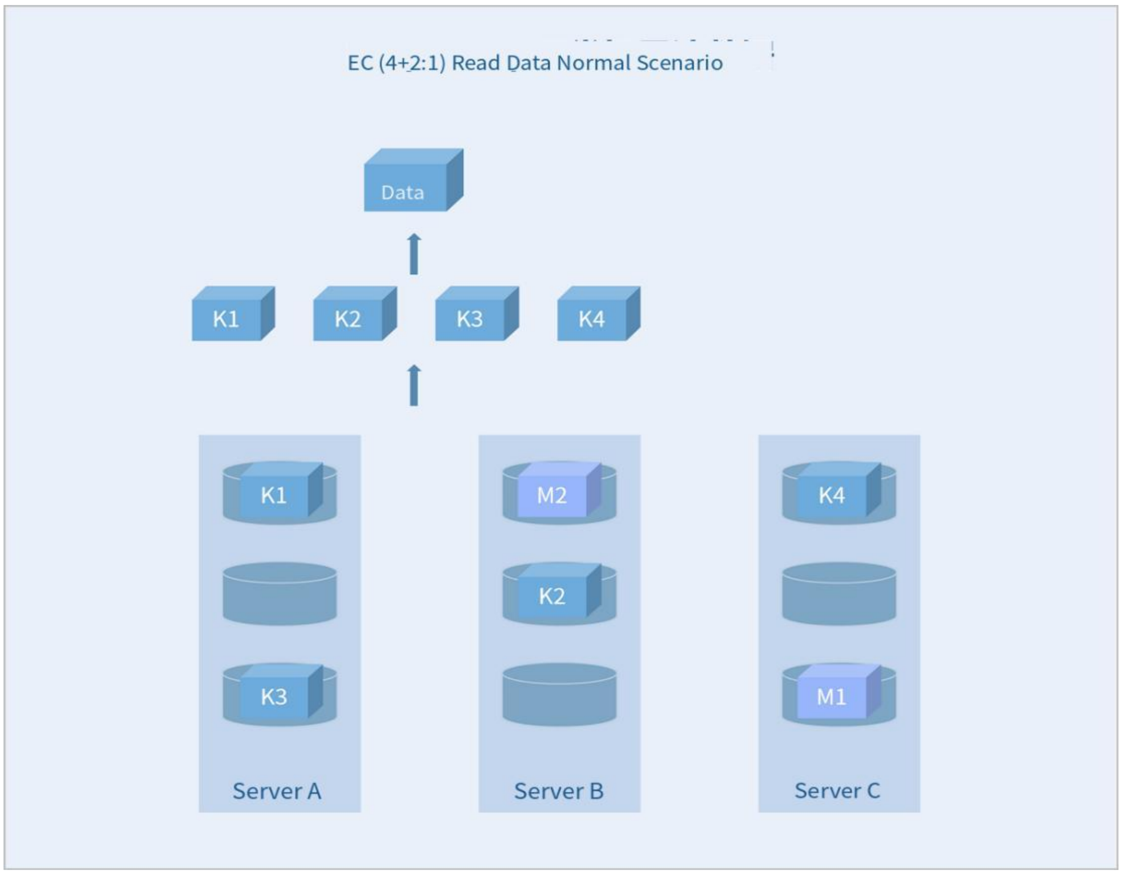


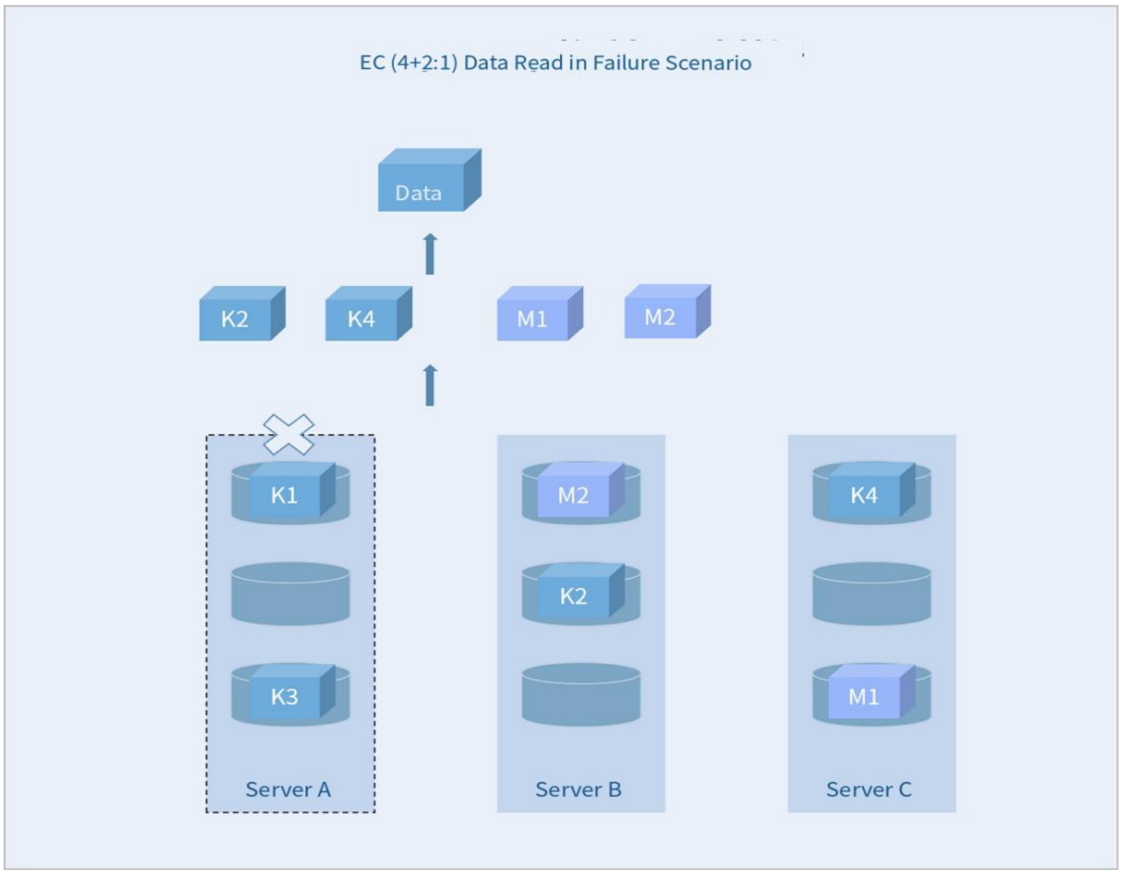
Figure 3-12: Stacked EC (4+2:1) Normal Read



- Read/write in fault scenario

Stacked EC (K+M:B): Using the server-level 4+2:1 EC strategy as an example, when a server or M disks fail, the system will still write the data into the remaining normal servers with K+M data fragments and check fragments. When reading data, the system reads data from other normal servers and recovers the data through the check algorithm before returning it to the user.

Figure 3-13: Stacked EC (4+2:1) Fault Read



Comparison between Replica and Erasure Code

According to your business requirements, you can choose replica or erasure code. These two strategies have their own advantages in different scenarios:

- **Space Efficiency:** The efficiency of erasure code is higher than that of replica. For example, the space efficiency of 4+2 EC strategy is about 66%, while that of 3x replica is 33.3%.

- **Read/Write Performance:** The read/write performance of replica and erasure code differs significantly under small data read/write scenarios, while the difference between these two strategies will gradually decrease under large data read/write scenarios. When data is written, erasure code involves data validation and may cause write penalties. In addition, because data is stored across multiple nodes; a high data latency of a node may seriously affect the performance. While reading data, replica only needs to read one object and does not involve data splicing across nodes.

- **Reconstruction Performance:** In general, the reconstruction performance of replica is better than that of erasure code, because replica reconstruction only involves simple data copying without data validation. While erasure code reconstruction involves a reverse validation computation, which consumes more read/write data and CPU computing resources.

- **Tolerance to Node Faults:** These two strategies have their own advantages. With regards to replica, multiple replica technologies allow your data to be stored without being lost if all but one of the non-monitored nodes fail. With regards to erasure code, for example, 4+2 EC strategy allows your data to be stored without being lost if two non-monitored nodes are failed.

3.2.3 Failure Domain Isolation

A failure domain is the minimum unit for distributing data in a cluster. When storing data, different copies or slices of a file will be stored in different failure domains based on the data redundancy policy. This way, even if some failure domains fail, data is not lost, ensuring data security. NexaVM nSDS provides three levels of failure domain isolation: server, rack, and data center.

- Server-level: Each server in the cluster is a failure domain. Different copies or slices of a file will be stored on different servers.
- Rack-level: Each rack in the cluster is a failure domain. Different copies or slices of a file will be stored on different racks. We recommend that you select this option if your cluster has a large scale and many racks.
- Data center-level: Each data center in the cluster is a failure domain. Different copies or slices of a file will be stored on different data centers. We recommend that you select this option if your cluster has a large scale and many data centers.

Figure 3-14: Server-level Failure Domain

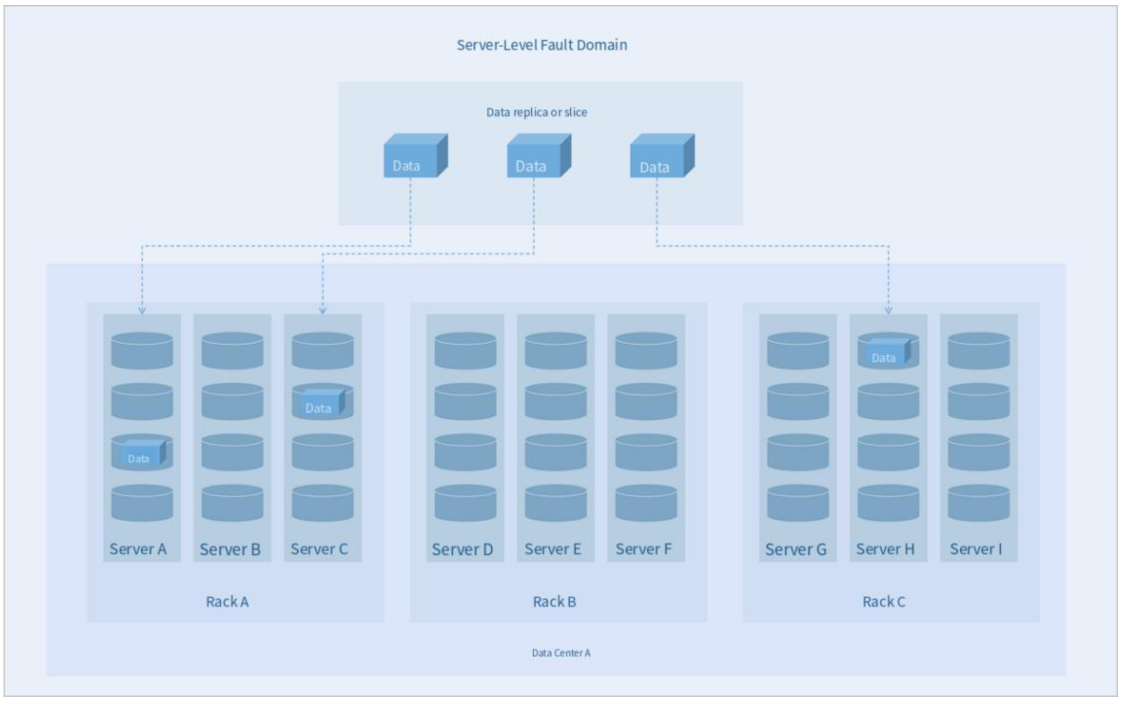


Figure 3-15: Rack-level Failure Domain

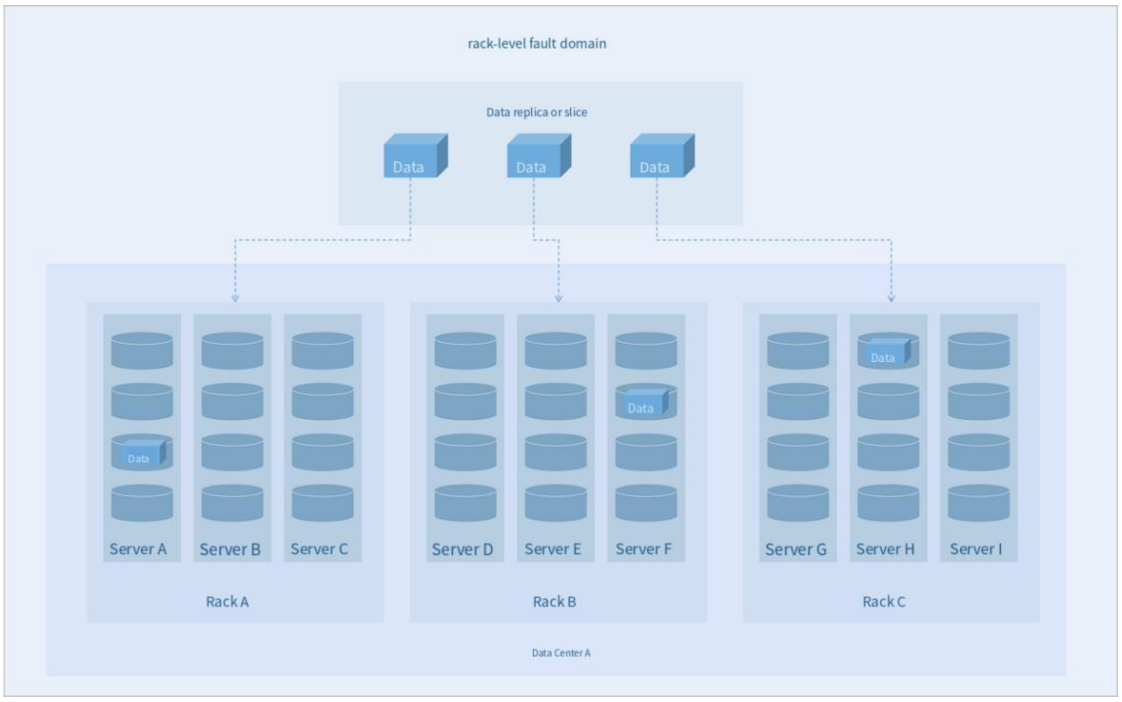
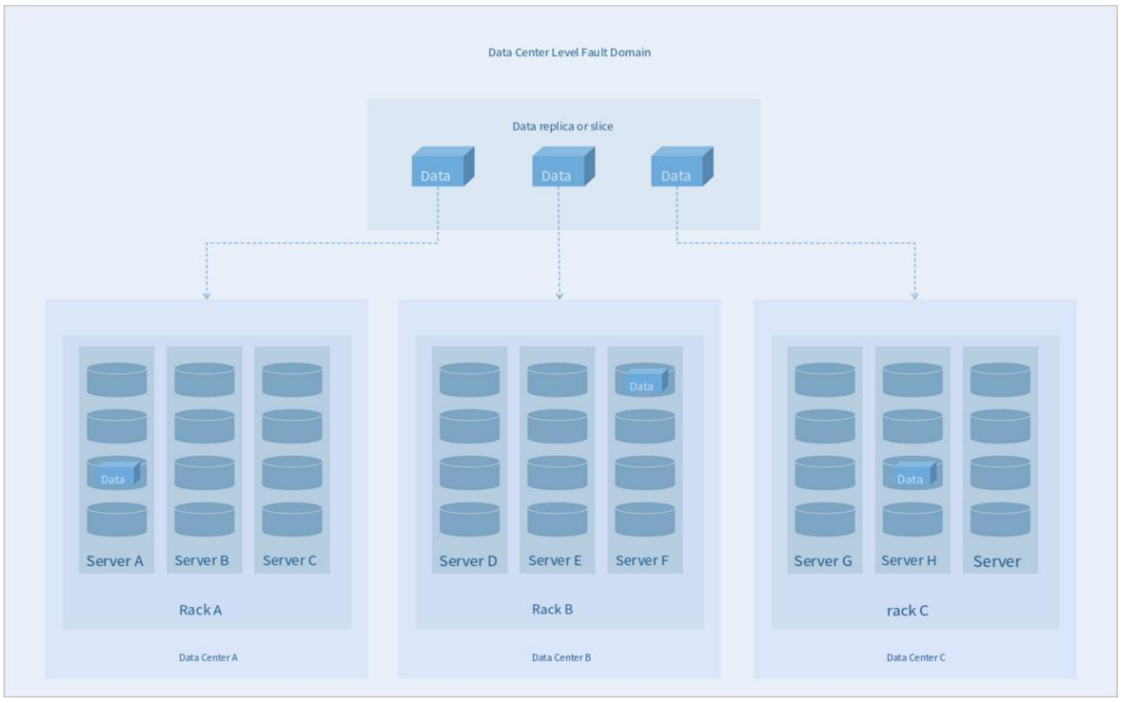


Figure 3-16: Data Center-level Failure Domain



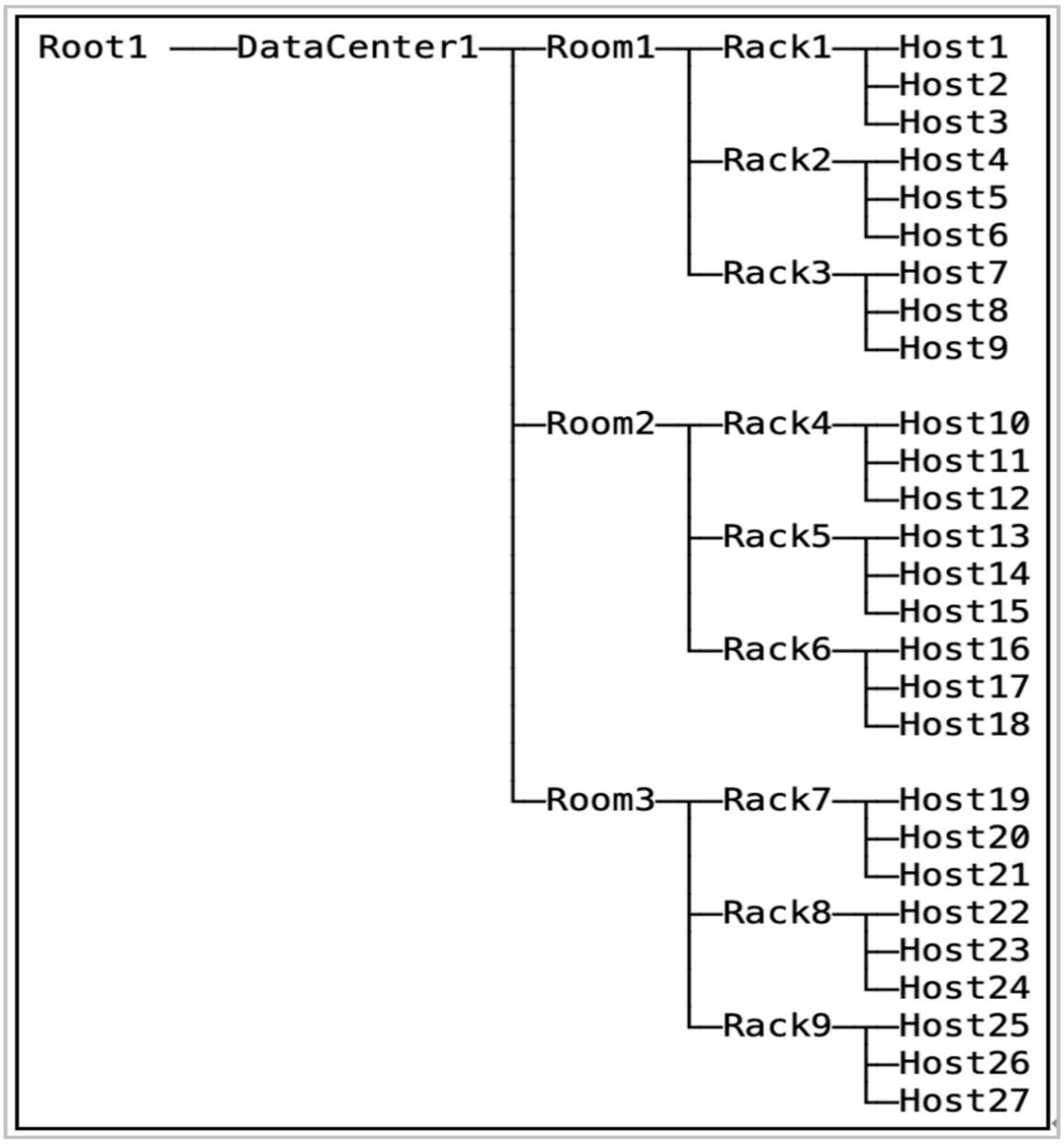
With failure domain isolation, the impact of failures is limited within a specific range, which avoids the situation where a small change causes wide-ranging effects. This improves business continuity.

With the failure domain expansion technology, you can add new nodes and make them an independent storage pool, which avoids data migration. This approach enables seamless scale- out, shielding application workloads from the underlying storage changes. This way, the workload of maintenance and business personnel is reduced, and system reliability and performance are improved.

3.2.4 Cluster Hardware Topology

A cluster hardware topology is a visualized display of the actual deployment of the cluster's physical resources, including data centers, rooms, racks, servers, and other logical entities. It describes the distribution relationship from rooms to servers through tree diagrams, with each tree diagram having a root node. Note that a cluster hardware topology can have multiple root nodes. You can select a data redundancy strategy corresponding to the topology level when you create a storage pool after the topology is planned.

Figure 3-17: Topology Hierarchy



You can plan the NexaVM nSDS topology on the Web interface. The following table lists the requirements on the number of each topology object.

Topology Object	Range
Data Center	0–2
Room	0–100 (in a datacenter)
Rack	0–100 (in a room)
Server	0–20 (in a rack)

3.2.5 Data Consistency Inspection

NexaVM nSDS applies to the Scrub mechanism to solve data consistency issues by scanning in the background. Data consistency inspections are periodic behaviors, divided into Scrub and Deep-Scrub.

- Scrub check: This check applies to metadata. It is characterized by a short execution time and a frequent schedule. We recommend that you perform consistency inspections daily. You can schedule the Scrub time as needed.
- Deep-Scrub check: This check applies to data. It is characterized by a long execution time and I/O pressure. We recommend that you perform consistency inspections during off-peak hours. If a Deep-Scrub check is not completed for more than 30 days, NexaVM nSDS issues a warning message.

3.3 Convenient O&M

NexaVM nSDS provides comprehensive, flexible, and convenient O&M features, including multiple resource pool management, hard disk lamp detection, disk S.M.A.R.T., data disk maintenance mode, data rebalancing, automatic fault detection and alarm, and more.

3.3.1 Multiple Resource Pools

NexaVM nSDS supports the multiple resource pool feature that allows you to use different types of storage media and isolate faults among them.

Each resource pool can have different attributes and performance, including replica count, data redundancy level, and storage medium. You can divide and manage your storage resources as required and improve storage efficiency and performance.

The resource pools are isolated from each other. You can manage different data on different resource pools. In addition, faults on a resource pool do not affect other resource pools, which effectively ensure data security and storage reliability.

3.3.2 Hard Disk Beacon

NexaVM nSDS allows you to virtualize beacons and quickly locate disks. If you need to maintain or replace a disk, you can enter the disk page of NexaVM nSDS and click the **Hard Disk Beacon** of the disk that needs to be maintained or replaced. Then, the LED of the corresponding disk in the physical environment is lit to guide you to quickly and accurately locate the device and improve maintenance efficiency.

3.3.3 Disk S.M.A.R.T. Detection

NexaVM nSDS supports S.M.A.R.T. technology to monitor health status, temperature, firmware, and data written of disks. Upper-layer businesses trigger alarms based on the I/O errors and disk status returned from the Smart Data.

3.3.4 Data Rebalancing

Data rebalancing distributes data evenly across the data volumes of all storage nodes in a storage cluster. This improves the performance and reliability of the storage system.

Automatic Data Rebalancing:

- NexaVM nSDS automatically rebalances databased on the storage pool settings and the workloads of the storage nodes. NexaVM nSDS moves data from a heavily loaded node to a lightly loaded node to achieve workload balancing.
- If a storage node in the storage cluster fails or a new storage node joins the storage cluster, NexaVM nSDS automatically migrates data to ensure data consistency and reliability.

Manual Data Rebalancing:

You can also manually rebalance data according to your specific needs.

3.3.5 Data Volume Maintenance Mode

NexaVM nSDS allows you to place a volume used for data storage in maintenance mode when you need to perform maintenance on a server or a disk. In maintenance mode, the volume stops responding to read and write requests and stops data balancing.

3.3.6 Automatic Fault Detection and Alarm

NexaVM nSDS supports automatic monitoring and alarm mechanisms that supervise the storage platform and various storage servers. Upon detecting failures, it automatically sends alarm messages to the platform. You can add email alarmers to receive the alarm messages and take measures for failure remediation.

When failures occur, NexaVM nSDS automatically restarts services and migrates data to ensure reliability and availability of data, thereby forming a highly available and highly reliable distributed storage system.

3.4 Product Integration

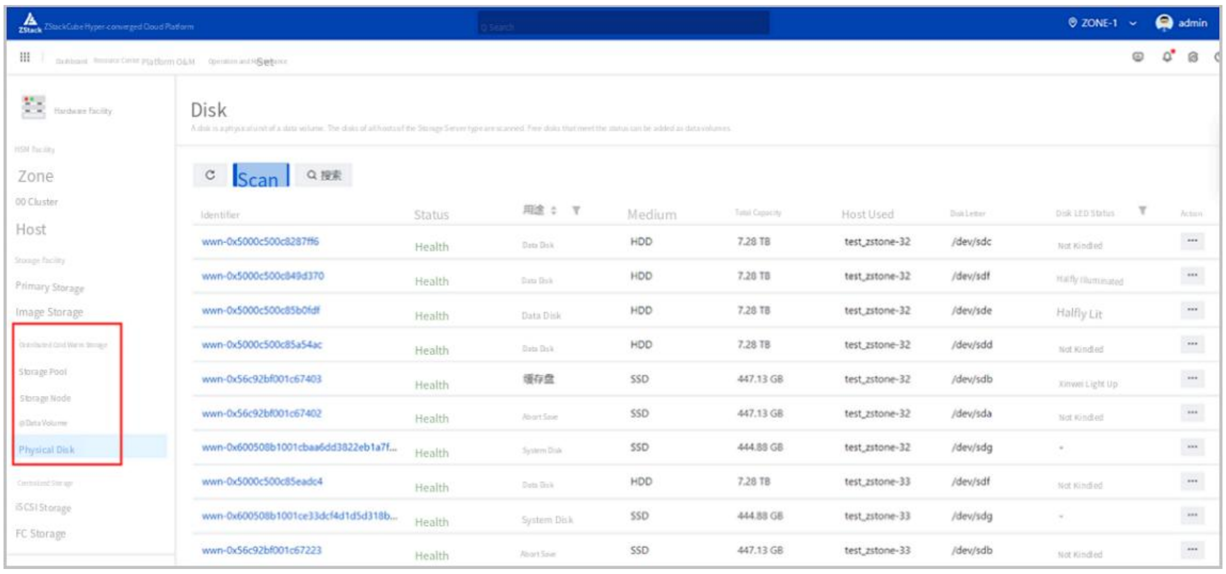
The NexaVM nSDS provides a RESTful management interface to allow seamless integration with upper-level applications.

3.4.1 NexaVM Cube Integration

As of 4.2.0, NexaVM nSDS supports the integration with NexaVM Cube, achieving unified monitoring and convenient scale-out for NexaVM Cube storage.

After the integration, you can one-click SSH login to NexaVM nSDS from the NexaVM Cube bootstrap or UI and can view and use the storage services provided by NexaVM nSDS and manage the storage resources in real time.

Figure 3-18: NexaVM Cube Integration with NexaVM nSDS



Disk

A disk is a physical unit of a data volume. The disks of all hosts of the Storage Server type are scanned. Free disks that meet the status can be added as data volumes.

Identifier	Status	用途	Medium	Total Capacity	Host Used	Disk Letter	Disk LED Status	Action
wwn-0x5000c500c8287ff6	Health	Data Disk	HDD	7.28 TB	test_zstone-32	/dev/sdc	Not Kindled	···
wwn-0x5000c500c849d370	Health	Data Disk	HDD	7.28 TB	test_zstone-32	/dev/sdf	Halfly Illuminated	···
wwn-0x5000c500c85b0fdf	Health	Data Disk	HDD	7.28 TB	test_zstone-32	/dev/sde	Halfly Lit	···
wwn-0x5000c500c85a54ac	Health	Data Disk	HDD	7.28 TB	test_zstone-32	/dev/sdd	Not Kindled	···
wwn-0x56c92bf001c67403	Health	缓存盘	SSD	447.13 GB	test_zstone-32	/dev/sdb	Xinwei Light Up	···
wwn-0x56c92bf001c67402	Health	Abort Save	SSD	447.13 GB	test_zstone-32	/dev/sda	Not Kindled	···
wwn-0x600508b1001cbaa6dd3822eb1a7f...	Health	System Disk	SSD	444.88 GB	test_zstone-32	/dev/sdg	-	···
wwn-0x5000c500c85eadc4	Health	Data Disk	HDD	7.28 TB	test_zstone-33	/dev/sdf	Not Kindled	···
wwn-0x600508b1001ce33dcf4d1d5d318b...	Health	System Disk	SSD	444.88 GB	test_zstone-33	/dev/sdg	-	···
wwn-0x56c92bf001c67223	Health	Abort Save	SSD	447.13 GB	test_zstone-33	/dev/sdb	Not Kindled	···

Note:

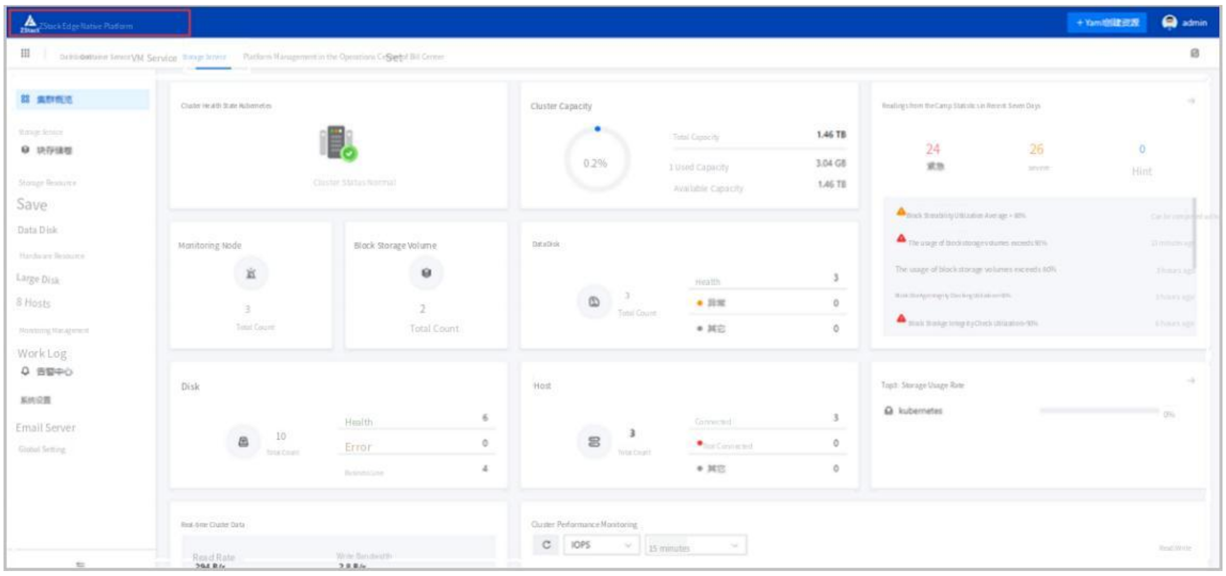
You can also view the integration progress in the **Integrations** page. By default, the status is displayed as **Integrating**. After the integration is completed, the status is displayed as **Normal**.

3.4.2 NexaVM Edge Integration

As of 4.2.2, NexaVM nSDS supports the integration with NexaVM Edge, achieving a unified monitoring of the NexaVM Edge storage layer.

After the integration, users can one-click SSH login to NexaVM nSDS via the NexaVM Edge bootstrap or UI and can directly view and use the storage services provided by NexaVM nSDS and real-time monitor and efficiently manage the storage resources.

Figure 3-19: NexaVM Edge Integration with NexaVM nSDS



3.4.3 Integrating a Third-Party Storage Platform

Starting from NexaVM nSDS 5.2.6, you can integrate a third-party storage platform to provide a unified interface for managing and monitoring resources across multiple platforms.

You can enable the Multiparadigm Addition option in Global Setting and complete the platform information configurations. After the integration, you no longer need to switch platforms frequently, which simplifies storage operations and improves management efficiency.

3.4.4 Virtualization and Cloud Native Platform Integration

NexaVM nSDS block storage service is mainly used to integrate virtualization platforms and cloud native platforms, providing virtual block devices for upper-layer businesses.

Virtualization Platform

NexaVM nSDS supports connecting to virtualization platforms to provide block devices for upper-layer virtual machines (VMs) as system volumes or data volumes. There are mainly three connection methods:

- By means of a kernel-space block storage interface:

The kernel-space block storage interface runs in the host operating system kernel space. This interface maps NexaVM nSDS virtual volumes to block devices in the host operating system. QEMU can use this interface to access the block devices.

The kernel-space block storage interface has limited features and has strong dependency on the host operating system kernel, which may cause high stability risks. We recommend that you do not use this method for production environments.

- By means of a user-space block storage interface:

The user-space block storage interface is a dedicated interface provided by NexaVM nSDS for accessing RBD block storage. It runs in the user space of the host operating system. QEMU can use this interface to access NexaVM nSDS block devices directly.

We recommend that you use this method to connect NexaVM nSDS to virtualization platforms and cloud platforms.

- By means of SPDK plus a user-space block storage interface:

SPDK is a high-performance user-space storage stack development library that works between QEMU and the user-space block storage interface. QEMU can access NexaVM NSDS block devices through the vhost socket file provided by SPDK.

SPDK uses kernel-bypass and I/O polling technologies to achieve high access speeds.

However, because SPDK-descended I/O is sent directly to hardware and I/O reaping is realized through polling hardware, it may cause hardware contention and lower I/O performance.

Cloud Native Platform

NexaVM nSDS supports connecting to cloud native platforms centered on Kubernetes to provide block devices for upper-layer container instances as persistent volumes.

The integration of NexaVM nSDS with cloud native platforms is mainly based on the CSI interface and the CSI plugin. CSI is a standard container storage interface, and the CSI plugin is an application that implements the CSI interface. The CSI plugin runs on multiple nodes of the cloud native platform and provides control services and volume lifecycle management. It can dynamically configure NexaVM nSDS block devices, map them to data volumes on the cloud native platform, and mount them to workloads to provide storage services.

4 File Storage Service

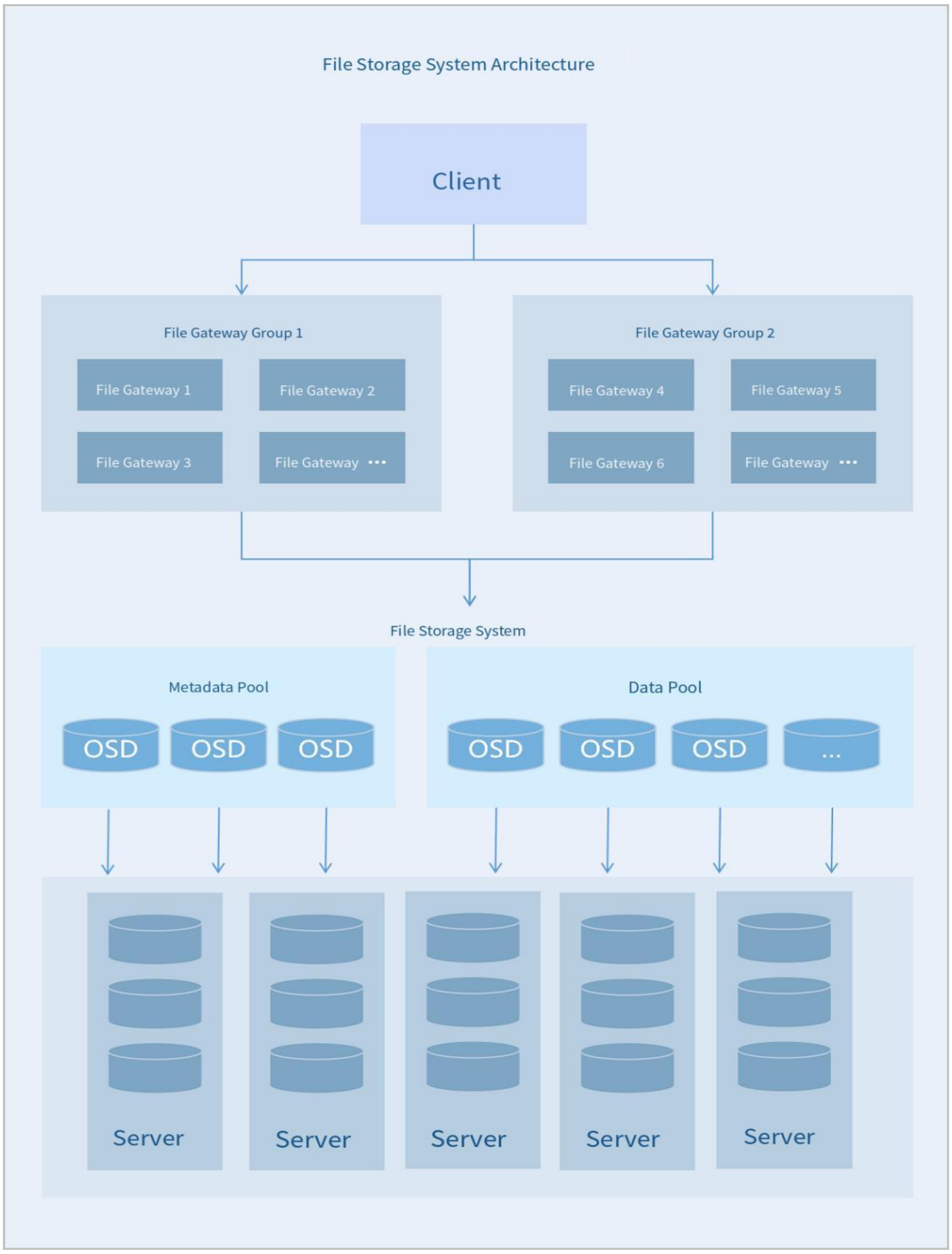
4.1 File System

File Storage System Architecture

NexaVM nSDS filesystems are built on top of Rados object storage and are managed by an MDS cluster. Metadata pools (Meta Pools) are used to manage and store metadata, while data pools (Data Pools) are used to manage data. Typically, metadata pools consist of a group of all-flash disks, and data pools can be configured as HDD pools or hybrid pools as needed.

After initializing a filesystem, NexaVM nSDS uses secondary subdirectories and the cluster public network to implement file sharing and supports accessing the file storage system via SMB/CIFS or NFS protocols. These shared resources are provided through file gateway groups, and you can access file storage systems via the public IP address of any gateway node. When writing or reading files, NexaVM nSDS first queries the metadata in the metadata pools via an MDS service and then directly accesses the data pools.

Figure 4-1: File Storage System Architecture



File Storage Functional Components

- File System: A filesystem is a core resource of a file storage service that is responsible for storing metadata and file data. Before you can use a file storage service, initialize a filesystem. This includes:

  ◦ Initialize MDS Service Cluster: Create an MDS service on each cluster monitoring node.

  ◦ Create File System: Create a distributed filesystem.

- File Directory: A file directory is a subset of a filesystem and is a must-have resource for providing file sharing and storing files and other resources.

- File User: A file user is a user of a file storage service. Currently, a file user in NexaVM nSDS is a local user for SMB shares. A local user is a must-have resource when you create an SMB file share. In addition, different file users can access different file shares. When you create a file user, NexaVM nSDS creates a Linux system user on all file gateway nodes in advance and adds the user to SMB shares.

- File Gateway: A file gateway is an entry that provides access to a file storage service. When accessing a file storage service, you need to specify one or more file gateways. You can create file gateway groups in NexaVM nSDS to manage multiple file gateways of SMB and NFS shares.

- File Share: A file share is a scenario of a filesystem that provides file storage services across multiple platforms. NexaVM nSDS provides SMB and NFS shares. SMB shares are mainly used for mounting Windows systems, while NFS shares are mainly used for mounting Linux systems. You can specify clients, client groups, or local users for a file share and set different permissions for these clients. Then, all other clients are not allowed to access the file share.

Scenarios

- File Share: Supports multiple sharing protocols to meet cross-platform sharing service requirements.
- Data Archiving: Provides sufficient storage space and a good data protection mechanism.
- Media Service: Is suitable for storing and accessing large-scale videos and images.

4.2 File Sharing

NexaVM nSDS allows you to share file directories by using the SMB protocol and the NFS protocol.

SMB Sharing

NexaVM nSDS realizes compatibility with Windows clients and other devices that support the SMB protocol by integrating Samba.

- **Implementation Method:** The VFS module in the Samba configuration file is used to mount NexaVM nSDS file storage services. By using this module, Samba can directly attach NexaVM nSDS filesystems to shared directories.

- **Typical Use:** Provides Windows file sharing support.

NFS Sharing

NexaVM nSDS uses NFS-Ganesha and the libcephfs interface of NexaVM nSDS to share file systems as NFS v3 and v4 protocols, which makes it easy for Unix/Linux clients to mount files.

- **Implementation Method:** Uses NFS-Ganesha and NexaVM nSDS 'libcephfs interface to provide NFS v3 and v4 protocol support.

- **Typical Use**

  ◦ Provides file sharing services by using NFS. This scenario applies to Linux-based clients.

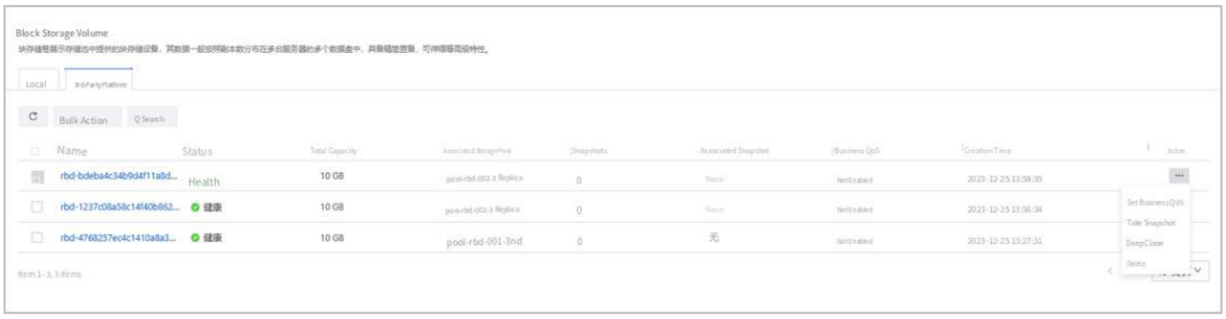
  ◦ Supports cross-platform data sharing, for example, sharing data across UNIX and Windows systems in a hybrid environment.

5 Block Storage Service

5.1 Third-Party LUN Take-Over

NexaVM nSDS allows you to take over third-party LUNs and synchronizes these LUNs nSDS, which are displayed separately from local volumes.

Figure 5-1: Third-Party LUN



Block Storage Volume

块存储卷展示存储池中提供的块存储设备，其数据一般按照副本数分布在多台服务器的多个数据盘中，具备精简置备、可伸缩等高级特性。

Name	Status	Total Capacity	Associated Storage Pool	Snapshots	Associated Snapshot	Business QoS	Creation Time	Action
rbd-bdeba4c34b9d4f11a8d...	Health	10 GB	pool-rbd-002-3 Replica	0	None	NotEnabled	2023-12-25 13:58:35	···
rbd-1237c08a58c14f40b862...	健康	10 GB	pool-rbd-002-3 Replica	0	None	NotEnabled	2023-12-25 13:56:34	
rbd-4768257ec4c1410a8a3...	健康	10 GB	pool-rbd-001-3nd	0	无	NotEnabled	2023-12-25 13:27:31	

Item 1-3, 3 items

5.2 Volume Migration

NexaVM nSDS allows you to migrate a block storage volume within the same cluster across different storage pools. The migration process includes the following three steps:

1. Prepare Migration: When a volume migration is launched, the Cloud creates a target volume and links the source volume to the target volume.

Similar to a thick volume, when data is read from uninitialized regions of the target volume, the mechanism redirects the read requests they make to the source volume. When data is written to uninitialized regions of the target volume, the Cloud performs deep copies to the target volume the overlapped blocks in the source volume.

2. Execute Migration: In the background, the Cloud performs deep copies to initialize all initialized blocks of the source volume to the target volume. This step can be executed concurrently while users are using the target volume.

3. Complete Migration: When the background migration is completed, you can commit or cancel the migration task. Committing a migration task deletes the cross-link between the source and target volumes and the source volume. Canceling a migration task deletes the cross-link and the target volume.

5.3 Volume Clone

NexaVM nSDS allows you to clone multiple block storage volumes based on a volume snapshot. Two types of cloning are provided: linked clone and independent clone.

A linked clone is implemented through a snapshot tree and Copy-on-Write (COW) technology. First, the system creates a read-only snapshot for the source volume. Then, the system uses COW to copy the snapshot and the original data to other clone volumes. These clone volumes share the same physical storage space. Data is written only when the data is changed. This mechanism saves storage space.

A linked clone has a dependency on the source snapshot. NexaVM nSDS provides a Disassociate button to remove the dependency and convert the linked clone volume into an independent volume.

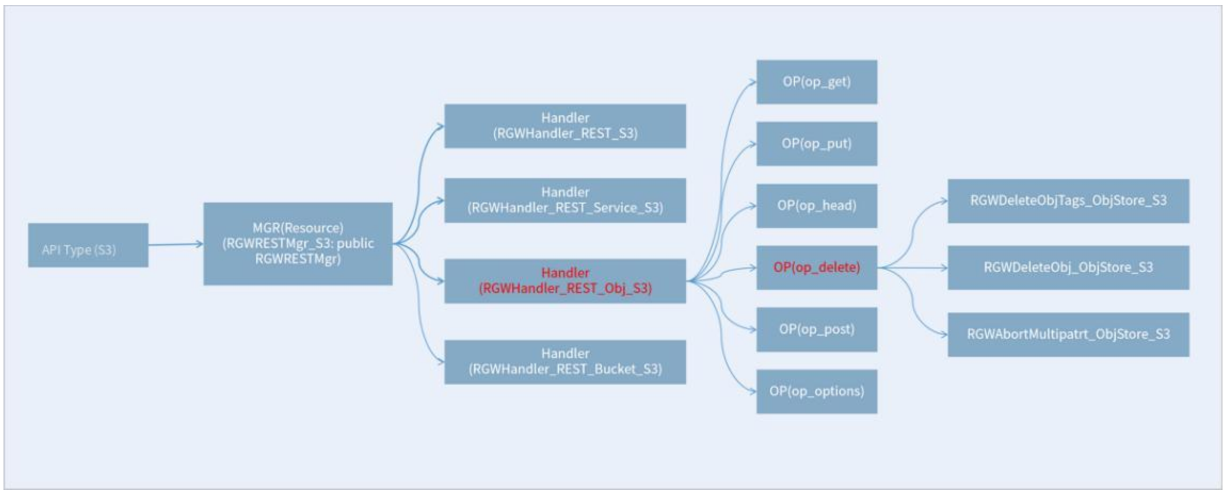
6 Object Storage Service

6.1 S3 Interface

NexaVM nSDS provides an interface that is compliant with the S3 protocol and is RESTful API. The interface provides object storage services for clusters.

Each API type corresponds to a primary management service, which can also register one or more secondary management services. Therefore, each API type might correspond to one or more management services. Each management service maintains the resources supported by the management service. You can look at each management service as a type of resource . Each resource might contain one or more handlers, and each handler might contain one or more operations (OP). The following figure uses the API type of S3 as an example to show the op_delete operation.

Figure 6-1: S3 API Type Relationship Diagram


API Type (S3)
MGR(Resource) (RGWRESTMgr_S3: public RGWRESTMgr)
Handler (RGWHandler_REST_S3)
Handler (RGWHandler_REST_Service_S3)
Handler (RGWHandler_REST_Obj_S3)
Handler (RGWHandler_REST_Bucket_S3)
OP(op_get)
OP(op_put)
OP(op_head)
OP(op_delete)
OP(op_post)
OP(op_options)
RGWDeleteObjTags_ObjStore_S3
RGWDeleteObj_ObjStore_S3
RGWAbortMultipatrt_ObjStore_S3

6.2 Access Control Management

NexaVM nSDS supports data transmission and access security through three permission levels: key pairs, user permissions, and bucket permissions.

User Permission Management

NexaVM nSDS allows admin to create multiple users for object storage services. Each user has isolated data and does not affect each other. To access the object storage service, authenticate by using the credentials of the user. After the authentication is succeeded, you can access the data in the object storage service.

The fundamental information for authentication includes Access Key and SecretKey. After a user is created in NexaVM nSDS, the system automatically generates a key pair for the user. When an S3 client accesses the object storage service, the system compares the key information in the client request with the key pair. If the information is consistent, the authentication succeeds.

NexaVM nSDS also allows you to manage the permissions and quotas of the created users:

- Permission management:

You can set permissions for the user to access the storage bucket and objects. By default, the user has three permissions: read, write, and delete.

- Read: A user with this permission can read the bucket ACL, list objects in a bucket, and download objects. This permission cannot be cleared.

- Write: A user with this permission can create a storage bucket, modify the bucket ACL, upload objects, and more.

- Delete: A user with this permission can delete a storage bucket and objects.

- Quota management:

◦ Storage buckets: You can set the maximum number of buckets a user can own.

- Capacity: You can set the maximum capacity that a user can store objects.
- Objects: You can set the maximum number of objects that a user can access or operate.

Note:
In addition to the user quota, you can set bucket quota for a user. That is, you can set the maximum capacity and object number that a user can use in a storage bucket.

Bucket Permission Management

A bucket permission includes bucket ACL, bucket policy, and user quota.

- Bucket ACL:

ACL is used to control the behavior permissions of users on buckets. NexaVM nSDS provides basic permissions for buckets and objects.

Permission	Permission on Visitor
READ	Read. Bucket: list objects in a bucket;Object: read object data
READ_ACP	Read ACL. Bucket: read ACL of a bucket;Object: read ACL of an object

Permission	Permission on Visitor
WRITE	Write. Bucket: create, delete, and overwrite objects in abucket; Object : none
WRITE_ACP	Write ACL. Bucket: modify ACL of a bucket; Object: modifyACL of an object
FULL_CONTROL	All of the above permissions

- Bucket policy:

A bucket policy is an access control list (ACL) expressed by using JSON. You can use a bucket policy to grant permissions to buckets and objects in the bucket. Only the bucket owner can associate a policy with a bucket. Permissions attached to a bucket apply to all objects owned by the bucket.

Note:
The size of a bucket policy is limited to 20KB.

- User quota:

The permissions of a bucket are inherited from the permissions of the user. If a user has a quota, the bucket quota is also applied to the bucket.

6.3 Multipart Upload

NexaVM nSDS allows you to upload large objects through a multipart upload mechanism. With this mechanism, you can upload large objects by uploading the objects in parts. Each part is a continuous piece of the original object. You can upload these parts in any order. If a part fails to be uploaded, you only need to re-upload this part, which does not affect the other parts. After all parts are uploaded, NexaVM nSDS arranges the parts in the correct order and makes them a large object.

Note:
Each part generated during a multipart upload occupies a quota. After you complete a multipart upload, only one quota is occupied.

The multipart upload mechanism of NexaVM nSDS has the following advantages:

- Improved upload throughput: With this mechanism, you can upload large objects through uploading these parts in parallel, which improves the upload throughput.
- Reduced network fault impact: Each part is a small object. If a network fault occurs, you can quickly retransmit these parts and reduce the impact of the network fault.
- Support for pausing or canceling an upload: If you pause an upload, the uploaded parts are not deleted. You can resume the upload from where it is paused. If you cancel an upload, the uploaded parts are deleted, and you need to upload the large object again.

6.4 Storage Policy

A storage policy is a set of rules about the allocation sources and data storage forms of resources in a storage bucket. It handles the mapping between a storage bucket and storage pools and allocates data of different storage classes to corresponding storage pools. NexaVM nSDS supports seven storage classes for each storage policy, including storageclass_0 to storageclass_6, where storageclass_0 is the default storage class. You can add storage classes when you create a storage policy or on the details page of a storage policy. Each storage class corresponds to an independent data pool, and you can specify a storage class to store object data in the corresponding data pool.

6.5 Data Compression

Data compression is a technique that applies compression algorithms to data for lossless or lossy compression, saving storage space, lowering storage costs, and improving storage efficiency. NexaVM nSDS applies the Snappy algorithm and supports online compression enablement on the Cloud.

Figure 6-2: Data Compression Process

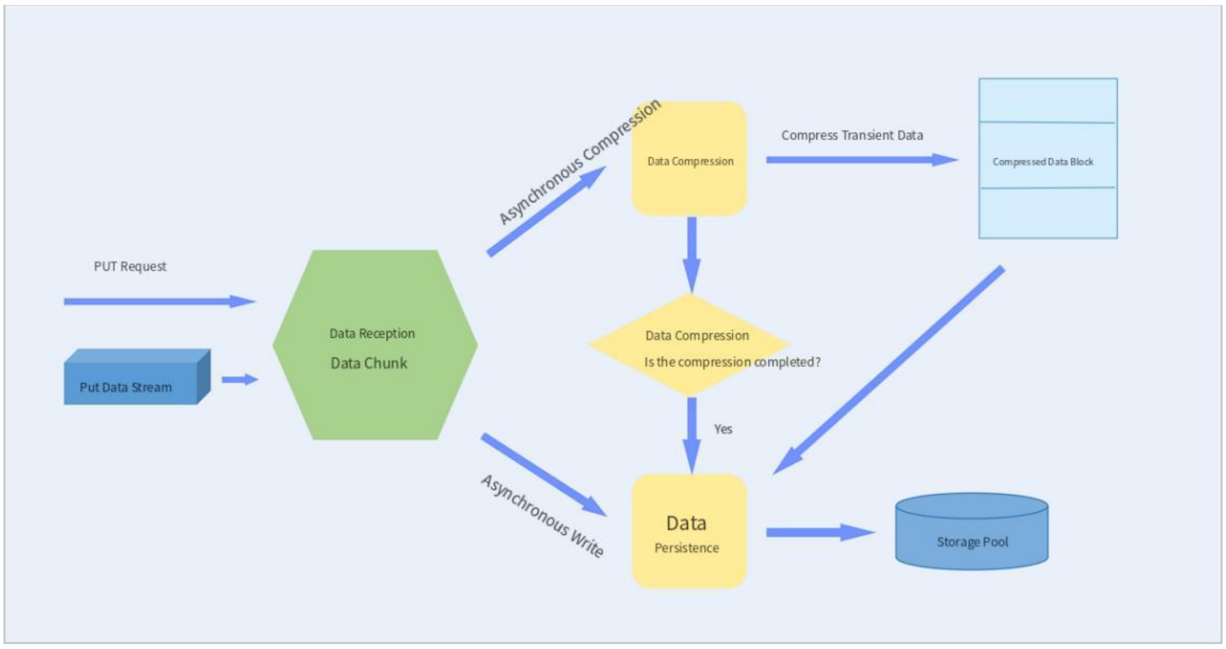
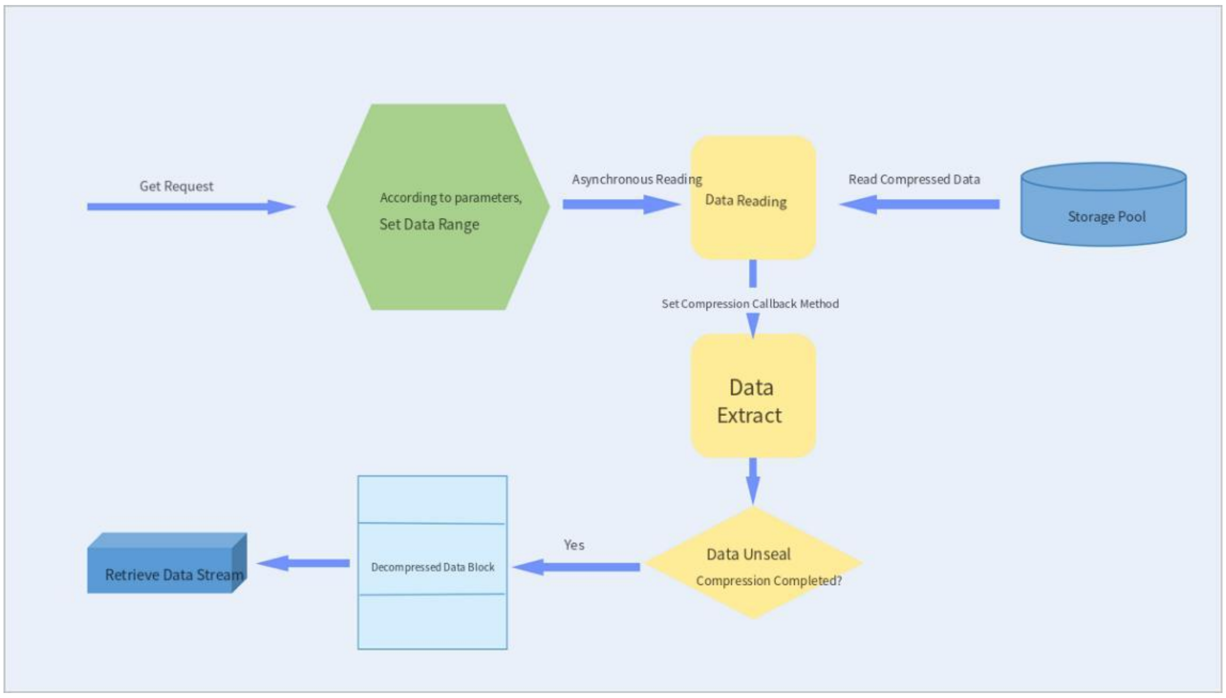


Figure 6-3: Data Decompression Process



7 Feature

NexaVM nSDS provides management and monitoring of multiple types of resources such as servers, hard disks, data disks, block storage volumes, storage pools, and storage buckets. This chapter introduces the features of NexaVM nSDS.

Table 7-1: Features

Type	Attribute	Description
Cluster	Cluster Overview	By Cluster Type: Displays the general storage cluster overview and high-performance storage cluster overview
		Displays the cluster storage status, cluster resource statistics, and the statistics of unread alerts in the last seven days in a card style
		Cluster Storage State: including cluster health status, cluster capacity stats, real-time data, and cluster performance monitoring (IOPS/Bandwidth/Delay), as well as capacity trend
		Cluster Resource Statistics: Includes Monitoring Node, Hardware Resources, Storage Resources, and Storage Service
File Storage	File Share	Supports two types of file protocols, including NFS and SMB, classified by protocol type
		Supports basic lifecycle management such as creating and deleting NFS file shares
		You can add up to 100 clients to an NFS file share
		Supports viewing NFS shared addresses and quickly obtaining NFS shared addresses, v3 recommended mount options, and v4 recommended mount options
		Supports basic lifecycle management such as creation and deletion of SMB file shares
		You can add up to 100 file users to a SMB file share
		Supports viewing SMB sharing addresses and quickly obtaining SMB sharing addresses and recommended mount parameters
	File Gateway	Allows you to create and delete file gateway groups and manage their basic lifecycles
		You can add or remove gateway nodes as per your business requirements. A file gateway group supports adding up to 10 servers as gateway nodes.

Type	Attribute	Description
	File User	Supports the basic lifecycle management of local file users, including creating, editing, and deleting file users
		Allows modifying local user passwords
	File Directory	Provides basic lifecycle management of file directories, such as creating and deleting file directories
	File System	Supports initializing filesystems, storing metadata information, and file data
Block Storage	Block Storage Volume	You can create and delete a volume and perform other basic lifecycle management operations on the local block storage
		Supports sync and other OM operations on third-party block storage volumes
		Allows you to set QoS for block storage volumes as per your business needs
		Supports expanding block storage volumes
		Supports migrating LUNs to other storage pools
		Allows you to create snapshots for block storage volumes, facilitating quick rollback in case of failures
		Allow rolling back a block storage volume to a specified snapshot point
		Support Deep Clone for Block Storage Volumes
	Snapshot Management	Supports centralized management of block storage volume snapshots, including cloning and deletion
		Support two cloning types: linked clone and independent clone
Object Storage	Storage Policy	You can create and delete storage policies to manage the lifecycle of resources
		You can set the default storage policy, compress data, and add storage tiers and other O&M operations
		You can add storage classes (storageclass_0 to storagecla ss_6) to meet the different storage requirements of objects in a storage bucket
	Object Gateway	Supports the basic lifecycle management of creating, enabling, disabling, and deleting S3 gateways
		Supports the basic lifecycle management of HA object gateways, including creation and deletion

Type	Attribute	Description
		You can add or remove load balancer listener nodes for HA object gateways
		Supports business roles and processes business requests for object gateways
	Object User	Supports the basic lifecycle operations (create, enable, disable, and delete) on object users
		You can set quotas for object users, including user quotas and bucket quotas
		Three user permissions are supported: read,write, and delete. You can customize your business requirements based on the permissions
		Supports modifying user quotas, adjusting user permissions, and other O&M operations such as modifying storage policies
		Supports the basic lifecycle management of generating and deleting user key pairs
	Storage Bucket	Supports the basic lifecycle management such as creating and deleting storage buckets
		You can set an account as the bucket owner. By default, the bucket owner has all permissions to the storage bucket
		You can grant specified ACL permissions (READ, WRITE, READ_ACP, WRITE_ACP, and FULL_CONTROL) to an object user
		Supports modifying bucket quotas and adding object users and other O&M operations
		Supports adding, removing, and modifying permission operations for authorized object users
Storage Resource	Storage Pool	You can create and delete a storage pool and perform other basic lifecycle management operations on it.
		Supports three types of storage pools: block storage, object storage, and file storage
		Supports four storage pool roles: data pool, index pool, composite pool, and metadata pool
		Supports two types of data redundancy: replica and EC
		Supports three levels of data redundancy: server, rack, and data center

Type	Attribute	Description
		Allows you to resize a storage pool, set the recovery QoS, and modify the replica number and other O&M operations
		After object storage initialization, object storage system resource pools can be displayed
		You can view the predicted capacity usage of the storage pool within 30 days
		Support Data-at-Rest Encryption for the VM Security using industry standard protocols like AES-256.
	Data Volume	Supports basic lifecycle operations such as creating and deleting data volumes
		Support setting data volume maintenance mode and other O&M operations
Hardware Resource	Disk	Supports scanning disks, setting cache partitions for disks (only for free disks), and other O&M operations such as setting disk indicator lights
	Topology	Supports visual management of cluster physical resource deployments
		Supports custom selection of topology objects to generate topology structures
		Supports canvas stretching, zooming in, and zooming out
	Server	Two types of servers are supported: storage server and storage gateway server
		A storage server provides hard disks and other resources for a storage pool. Hard disks on the server can be used as data disks
		A storage gateway server serves as interfaces for different clients. You can only manage the gateway server. Note that you cannot manage the disks in the server.
		Support five server roles: management, monitoring, block storage gateway, object storage gateway, and file storage gateway
		A management role is responsible for collecting and managing the cluster running state. It functions as a management node to provide distributed storage cluster management capabilities, supporting GUI and API management methods
		The monitoring role monitors the storage data for the cluster and maintains the cluster's overall state, including metadata information such as data mappings and cluster authentication

Type	Attribute	Description
		The LUN role of the block storage gateway is used to access the storage cluster via the Block interface.
		The object storage gateway role is responsible for accessing the storage cluster by using the Object interface with the server
		Enables S3 roles of the object storage gateway server to provide the S3 protocol and gateway service
		The server for the file storage gateway role supports creation of file gateways and provides access protocols such as SMB and NFS
		Servers of the storage server type support setting these five roles
		The server of the Storage Gateway server type can only be set as Block Storage Gateway role
		You can add or remove roles for the storage server
		Supports basic lifecycle management, including adding and deleting servers
	Cluster	Displays information of a generic storage cluster, including the cluster name, network configurations, time server address, and the number of servers
		Supports initializing high-performance storage clusters and setting RDMA transmission
Monitor Management	Operation Log	You can view the description of the operation, task status, operator, and time created/finished of the historical operations. In addition, you can view detailed responses of the operations. Thisachieves fine-grained management
	Alarm Center	Displays and centrally manages platform alarm messages in a list, improving O&M efficiency
		Supports screening by time period, allowing you to view alarm messages triggered within the selected period
		Support Mark Alarm Messages as Read
		Supports viewing alarm message details
		Support alarm messages sorted by alarm level (emergency, critical, warning) and alarm time
		Displays alarm rules predefined by alarmers in a list

Type	Attribute	Description
		Supports adding or removing endpoints for alarms
		System notification endpoints are provided by the Cloud. If you use System as endpoints for alarms, you will receive messages sent from the Cloud
		Supports basic lifecycle management, including creation and deletion of a custom endpoint
System Settings	Email Server	Supports basic lifecycle management of email servers such as adding and deleting
		Email Server Type: smtp
		Support Testing Email Server Connectivity
		Allows setting an encrypted connection for the port of an email server, including encryption types such as STARTTLS/NONE and SSL/TLS
	Global Setting	Provides platform-level feature settings. After being set, these settings take effect globally on the platform
License Management	/	The distributed storage platform supports multiple license types . You can select a suitable license as needed.
		Support license types by large category: Platform License and Feature License
		Uses a platform license with functional licenses to provide comprehensive distributed storage functionalities that satisfy most of your business needs
		Supported platform license type: Enterprise Plus and Paid Enterprise
		A feature license works with a platform license to provide specific storage capabilities that meet your specific business requirements
		Supported feature license: block storage feature license, object storage feature license, high-performance block storage feature license, file storage feature license, and high-speed network feature license
		Supporting authorization methods: USB Key and Request Code
		Supports viewing the current license and historical authorization records

Type	Attribute	Description
		Supports license abnormal reminders (license expired, license quota exhausted, and license due)

Glossary

File Storage
The file storage function helps to store and manage the file data.

File Share
File share is the sharing of file directories through file protocols, allowing clients to map directories to their local filesystems through mount operations, enabling file sharing and collaboration.

File Gateway
The file gateway group provides users with file storage access protocols such as SMB and NFS.

File User
The file user is the account that uses the file storage service, containing authorization information such as the username and password required for file protocol access.

File Directory
The file directory provides users with management functions for first-level subdirectories within the filesystem, such as creating directories, deleting directories, searching for directories, and more.

File System
The filesystem is the core resource for file storage, responsible for storing metadata information and file data.

Block Storage
The block storage function provides high-performance service for important business.

Block Storage Volume
A block storage volume displays the block storage device provided in a storage pool. The data of a block storage volume is distributed in multiple data disks of multiple servers based on specified replicas and has advanced characteristics such as thin provisioning and scalability.

Snapshot
A snapshot is a point-in-time capture of data status in a block storage volume and provides data backup and recovery features for the storage system.

Object Storage
The object storage function delivers secure and efficient storage and management service for massive amounts of unstructured data.

Storage Policy
A storage policy is a set of rules governing resource allocation sources for storage buckets, data storage formats, and more.

Object Gateway
The object gateway consists of high-availability object gateways and S3 gateways. The S3

gateway provides an S3-compatible object storage service for accessing storage clusters, while the high-availability object gateway offers high-availability services and load balancing for object storage.

Object User
An object user is an account for a consumer of object storage services, containing information such as permissions, key pairs, and user quotas.

Bucket
A bucket is a logical storage space allocated to an object user, where user data is stored in the form of objects.

Storage Resource
A logical unit that provides storage services externally, consisting of a storage cluster.

Storage Pool
A storage pool is a logical partition in a storage cluster, which consists of storage severs and data disks to store objects.

Data Disk
A data disk is a logical storage unit, with each data disk corresponding to one data process. Multiple data disks can form a storage pool based on replication or Erasure Coding (EC) mechanisms.

Hardware
The hardware refers to a collection of hardware devices that form a storage cluster, such as servers, hard disks, and networking equipment.

Server
Servers are divided into storage servers and storage gateway servers.

Hard Disk
A hard disk is the physical unit of a data disk. All hard disks on storage servers are scanned and displayed in the list. Healthy free disks can be added as data disks.

Topology
Topology represents a visual depiction of how physical resources are actually deployed within a cluster. You can use this feature to configure data redundancy policies for storage pools.

Cluster
A cluster is a logical collection of servers.

Monitoring Management
The monitoring management is a monitoring system of the storage platform, consisting of operation log and an alarm center. The system helps you track the platform's operational status, quickly pinpoint issues and resolve them.

Operation Log
An operation log is a chronological record of operations on the specified resources and their results, which is convenient for troubleshooting and tracing. You can easily get the operation status of the whole platform.

Alarm Center
Alarm Center monitors time-series data and events, displays all alarm messages of the platform, and sends alarm messages to specified endpoints in real time according to their emergency level, to facilitate timely tracking of resource usage.

Alarm Message
An alarm message is a message sent the time when an alarm is triggered. Alarm Center displays all alarm messages of the platform.

Alarm
An alarm is used to monitor the status of time-series data and events and respond to the status change.

Endpoint
An endpoint is a method that users obtain subscribed messages. Endpoints are categorized into system endpoints and email.

System Setting
System setting consists of email server, theme setting, and global setting.

Email Server
If you select Email as the endpoint of an alarm, you need to add an email server. Then alarm messages are sent to the email server.

Theme Setting
You can customize the theme and appearance of the platform.

Global Setting
Global Setting allows you to configure settings that take effect on the whole platform.