

**Huawei OceanStor Pacific Series
8.2.0**

Technical White Paper

Issue 01
Date 2024-01-30



Copyright © Huawei Technologies Co., Ltd. 2024. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: <https://e.huawei.com>

Security Declaration

Product Lifecycle

Huawei's regulations on product lifecycle are subject to the *Product End of Life Policy*. For details about this policy, visit the following web page:

<https://support.huawei.com/ecolumnsweb/en/warranty-policy>

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process*. For details about this process, visit the following web page:

<https://www.huawei.com/en/psirt/vul-response-process>

For vulnerability information, enterprise customers can visit the following web page:

<https://securitybulletin.huawei.com/enterprise/en/security-advisory>

Initial Digital Certificate

The Initial digital certificates on Huawei devices are subject to the *Rights and Responsibilities of Initial Digital Certificates on Huawei Devices*. For details about this document, visit the following web page:

<https://support.huawei.com/enterprise/en/bulletins-service/ENEWS2000015789>

Huawei Enterprise End User License Agreement

This agreement is the end user license agreement between you (an individual, company, or any other entity) and Huawei for the use of the Huawei Software. Your use of the Huawei Software will be deemed as your acceptance of the terms mentioned in this agreement. For details about this agreement, visit the following web page:

<https://e.huawei.com/en/about/eula>

Lifecycle of Product Documentation

Huawei after-sales user documentation is subject to the *Product Documentation Lifecycle Policy*. For details about this policy, visit the following web page:

<https://support.huawei.com/enterprise/en/bulletins-website/ENEWS2000017761>

Contents

1 Overview..... 1

1.1 Challenges Faced by Storage Systems..... 1

1.2 OceanStor Pacific Series Highlights..... 2

1.2.1 Fully Symmetric Scale-out Architecture, Enabling Ultimate Elastic Scalability..... 2

1.2.2 Interworking of Unstructured Data Storage Services for On-Demand Use..... 2

1.2.3 Superb Capacity and Performance Density Design..... 3

2 Hardware Architecture..... 4

2.1 Hardware Introduction..... 4

2.1.1 Dedicated Storage Hardware..... 5

2.1.1.1 OceanStor Pacific 9950..... 5

2.1.1.2 OceanStor Pacific 9546..... 6

2.1.1.3 OceanStor Pacific 9550..... 7

2.1.1.4 OceanStor Pacific 9920..... 8

2.1.1.5 OceanStor Pacific 9520..... 8

2.1.1.6 OceanStor Pacific 9350..... 9

2.1.1.7 OceanStor Pacific 9346..... 9

2.1.1.8 OceanStor Pacific 9150..... 10

2.1.1.9 OceanStor Pacific 9146..... 11

2.1.2 Universal Storage Hardware..... 12

2.1.2.1 OceanStor Pacific 9540..... 12

2.1.2.2 OceanStor Pacific 9340..... 13

2.2 High-Performance Hardware Design..... 13

2.2.1 Front- and Back-End PCIe 4.0..... 13

2.2.2 Huawei-Developed Half-Palm NVMe SSD..... 14

2.2.3 Efficient Heat Dissipation Design..... 15

2.3 Large-Capacity Hardware Design..... 16

2.3.1 High-Cohesion Structure Design..... 16

2.3.2 Efficient Heat Dissipation Design..... 18

3 Software Architecture..... 19

3.1 Fully Symmetric Scale-out Software Architecture..... 19

3.1.1 Logical Software Architecture..... 19

3.1.1.1 Access Layer..... 20

3.1.1.2 Service Layer.....	20
3.1.1.3 Index Layer.....	21
3.1.1.4 Persistence Layer.....	22
3.1.1.5 Microservice Forms.....	22
3.1.2 Logical Network Architecture.....	23
3.2 Global Load Balancing.....	26
3.2.1 Front-End Load Balancing.....	26
3.2.1.1 Basic Routing Model.....	26
3.2.1.2 Route Mappings for File Semantics.....	27
3.2.1.3 Route Mappings for Block Semantics.....	28
3.2.2 Back-End Load Balancing.....	29
3.2.2.1 STORE DHT.....	30
3.2.2.2 Dynamic Intelligent Partitioning and Static Disk Selection Algorithms.....	30
3.3 Multi-Level Cache Acceleration.....	32
3.3.1 Write Cache Acceleration.....	33
3.3.2 Read Cache Acceleration.....	35
3.4 Key I/O Processes.....	37
3.4.1 Key I/O Processes of Unstructured Data Storage Services.....	37
3.4.2 Key I/O Processes of the Structured Data Storage Service.....	39
3.5 Convergence and Interworking of Unstructured Data Storage Services.....	39
3.5.1 Lossless Semantic Interworking.....	42
3.5.1.1 ListObjects Interface of the S3 Protocol.....	42
3.5.1.2 Multipart Interface of the S3 Protocol.....	44
3.5.1.3 Versioning Interface of the S3 Interface.....	45
3.5.2 Permission Interworking.....	46
3.5.2.1 NFS and SMB.....	46
3.5.2.2 S3 and Other Protocols.....	50
3.5.2.3 Domain Users.....	52
3.5.3 Lock Interworking.....	54
4 Value-added Feature: Efficiency Improvement.....	57
4.1 SmartMulti-Tenant (Multi-tenancy).....	58
4.2 SmartQuota (Quota).....	59
4.3 SmartTier (Tiered Storage).....	62
4.4 SmartEqualizer (Load Balancing).....	64
4.5 SmartIndexing (Metadata Indexing).....	65
4.6 SmartTakeover (Intelligent Takeover).....	66
4.7 SmartQoS (Intelligent Quality of Service Control).....	67
4.7.1 Unstructured Data Storage Services.....	67
4.7.2 Structured Data Storage Service.....	69
4.8 SmartAuditlog (Audit Log).....	71
4.9 SmartEncryption (Data Encryption).....	77
4.9.1 Software and Hardware-integrated Encryption.....	77

4.9.2 SED.....	78
4.10 SmartDedupe and SmartCompression (Deduplication and Compression).....	80
4.11 SmartMove (Online Volume Migration).....	82
4.12 VMware Virtual Volume (vVol).....	86
4.13 Scenario-specific SmartCompression (Scenario-specific Compression).....	87
4.14 Standard SmartCompression (Standard Compression).....	88
4.15 SmartMigration (Intelligent Data Migration).....	90
4.15.1 File Migration Service.....	90
4.15.2 Volume Migration Service.....	92
4.15.2.1 Online Takeover.....	93
4.15.2.2 Offline Takeover.....	96
4.15.2.3 Online Migration of Heterogeneous Volumes.....	97
4.16 SmartSync (Copying Objects to the Cloud).....	100
5 Value-added Feature: Data Protection.....	102
5.1 HyperSnap (Snapshot).....	102
5.1.1 Unstructured Data Storage Services.....	103
5.1.2 Structured Data Storage Service.....	104
5.2 HyperReplication (Remote Replication).....	105
5.3 HyperMetro (Active-Active).....	110
5.4 HyperGeoMetro (Cross-Site Multi-Active of the Object Service).....	111
5.5 HyperGeoEC (Cross-Site EC of the Object Service).....	112
5.6 HyperClone (Linked Clone).....	112
5.7 Recycle Bin.....	113
5.8 Antivirus.....	114
5.9 HyperLock (WORM).....	115
5.10 NDMP Backup.....	116
6 End-to-End Performance Optimization.....	119
6.1 FlashLink.....	123
6.1.1 Data-Control Separation.....	123
6.1.2 Intelligent Multi-Core Technology.....	124
6.1.3 ROW Full-Stripe Write.....	124
6.1.4 Intelligent Stripe Aggregation.....	125
6.1.5 End-to-End I/O Priority.....	125
6.2 RDMA High-Speed Network.....	127
6.3 File Service Parallel Client.....	127
6.4 Distributed Lock-Free Design.....	130
6.5 Mixed I/O Load Optimization.....	130
6.6 GPU Data Read/Write Acceleration.....	131
6.7 AI Cache.....	131
7 System Reliability Design.....	134
7.1 Module-Level Reliability.....	134

7.1.1 Disk Reliability.....	135
7.1.2 HSSD Reliability.....	136
7.1.3 NIC Reliability.....	136
7.1.4 Memory Reliability.....	136
7.2 Node-Level Reliability Design.....	137
7.2.1 Power Failure Protection.....	137
7.2.2 Link Reliability.....	137
7.2.3 Node Self-Healing Protection.....	138
7.2.3.1 High Temperature Protection.....	138
7.2.3.2 Hardware Self-Healing Protection.....	139
7.2.4 Node Failover.....	139
7.3 System-Level Reliability.....	140
7.3.1 Data Reliability Design.....	140
7.3.1.1 Data Redundancy Protection.....	140
7.3.1.1.1 Data Security Layout Policy.....	140
7.3.1.1.2 Erasure Coding (EC).....	141
7.3.1.1.3 Never Degraded Data Writes.....	142
7.3.1.1.4 Fast Data Rebuilding.....	143
7.3.1.1.5 Dynamic Increase of EC Fragments.....	145
7.3.1.2 Data Integrity Protection.....	145
7.3.1.2.1 I/O Path Data Integrity Protection.....	146
7.3.1.2.2 Periodic Check of Persistent Data.....	147
7.3.1.2.3 Correction and Self-Healing of Corrupted Data.....	148
7.3.1.3 Restoration of Mistakenly Deleted Data.....	148
7.3.2 Sub-health Management.....	148
7.3.2.1 Disk Sub-health Management.....	148
7.3.2.2 Network Sub-health Management.....	148
7.3.2.3 Service Sub-health Management.....	149
7.3.2.4 Fast-Fail Mechanism.....	151
7.4 Solution-Level Reliability Design.....	152
7.4.1 Local Data Protection.....	152
7.4.2 Site-Level Data Protection.....	153
8 System Serviceability Design.....	154
8.1 System Management Tools.....	154
8.1.1 DeviceManager.....	155
8.1.2 CLI.....	156
8.1.3 RESTful API.....	156
8.1.4 SNMP.....	157
8.1.5 SmartKit.....	157
8.1.6 eSight.....	157
8.1.7 eService.....	158
8.2 Resource Management for Storage Services.....	158

8.2.1 Resource Management for the File Service.....	158
8.2.2 Resource Management for the Object Service.....	159
8.2.3 Resource Management for the HDFS Service.....	160
8.2.4 Resource Management for the Block Service.....	160
8.3 System O&M Management.....	161
8.3.1 Node Addition and Deletion.....	161
8.3.2 System Upgrade.....	161
9 System Security Design.....	163
9.1 Storage Device Security.....	164
9.1.1 Operating System Hardening.....	164
9.1.2 Secure Startup.....	164
9.1.3 Patch Management.....	165
9.2 Storage Network Security.....	165
9.2.1 Plane Isolation.....	165
9.2.2 Channel Security.....	165
9.3 Storage Service Security.....	166
9.3.1 Access Control.....	166
9.3.1.1 File Service Access Control.....	166
9.3.1.2 Object Service Access Control.....	167
9.3.1.3 HDFS Service Access Control.....	168
9.3.1.4 Block Service Access Control.....	168
9.3.2 Data Encryption.....	168
9.3.3 Service Access Log Audit.....	171
9.3.4 WORM.....	171
9.4 Storage Management Security.....	172
9.4.1 Authentication.....	172
9.4.2 Role Management.....	173
9.4.3 Log and Alarm Management.....	173
9.4.4 Web Security.....	173
9.4.5 User Security.....	174
9.4.6 Password Security.....	175
10 Ecosystem Compatibility.....	178
10.1 Ecosystem Compatibility of the Data Plane.....	178
10.1.1 Compatibility with Storage Protocols.....	178
10.1.2 Block Service Compatibility with Virtualization Platforms.....	178
10.1.3 Block Service Compatibility with Databases.....	179
10.1.4 Block Service Compatibility with Operating Systems.....	179
10.1.5 File Service Compatibility.....	179
10.1.6 Object Service Compatibility.....	179
10.1.7 HDFS Service Compatibility.....	180
10.2 Ecosystem Compatibility of the Management and Control Plane.....	180
10.2.1 Integrated Network Management Platforms.....	180

10.2.2 OpenStack Integration..... 180

10.2.3 Compatibility with Container Platforms..... 180

11 Mappings Between Features and Services.....181

12 Abbreviations, Acronyms, and Glossary..... 183

1 Overview

[1.1 Challenges Faced by Storage Systems](#)

[1.2 OceanStor Pacific Series Highlights](#)

1.1 Challenges Faced by Storage Systems

With the advent and popularization of cloud, Internet, and intelligent technologies, new enterprise applications are constantly emerging and data volumes increase rapidly. Accelerating service innovation poses higher requirements on fast iteration of storage capabilities. For example, the finance industry is facing a host of new opportunities and challenges brought by e-Banking and mobile Internet finance in particular. Such challenges include needs for precise user requirement analysis and hour-level service rollout periods. A surge in new services and an exponential increase in service data can be seen in the fields of smart city, smart manufacturing, and telecom carriers. These developments pose the following new challenges for storage systems in enterprise data centers:

- Tension between long storage system construction periods and short rollout periods of new services
- Huge system size and complex management, exerting high pressure on operation and maintenance (O&M) personnel
- Inability of storage systems to meet increasing concurrent data processing requirements
- Higher demand for big data and cloud computing technologies that facilitate customer requirement analysis, service data analysis, and decision making

New challenges will inevitably raise new requirements. If you are faced with the preceding challenges, then your ideal storage system may be something like this:

- The system is agile. Resources can be deployed flexibly and acquired on demand. The rollout periods of new services are shortened.
- Enriched access methods are provided for diverse types of data in data centers, such as structured and unstructured data.
- Implementing quick and large-scale expansion of performance and capacity is a piece of cake.

- Superb performance is delivered to process data concurrently.
- The total cost of ownership (TCO) is decreased.

If this is what you are looking for, Huawei OceanStor Pacific series is the answer. As a data center-class product, it provides intelligent scale-out storage and supports elastic large-scale scale-out. By using system software, OceanStor Pacific series organizes local storage resources of storage nodes to build scale-out storage pools and provide scale-out file, object, Hadoop Distributed File System (HDFS), and block services for upper-layer applications, as well as a variety of service functions and value-added features. It leverages unique elastic erasure coding (EC) technology to provide more available storage space while ensuring service performance. You can purchase and deploy storage services based on service requirements to help enterprises easily cope with ever-changing services and implement flexible, efficient, and on-demand data access.

1.2 OceanStor Pacific Series Highlights

1.2.1 Fully Symmetric Scale-out Architecture, Enabling Ultimate Elastic Scalability

OceanStor Pacific series adopts a fully symmetric scale-out architecture. It enables a linear growth in system capacity and performance by increasing storage nodes, requiring no complex resource requirement plans. It can be easily expanded to contain thousands of nodes and provide EB-level storage capacity. This helps meet your future storage demands. The system evenly distributes data and metadata on nodes and automatically balances loads, eliminating metadata access bottlenecks and ensuring system performance after capacity expansion. The system leverages FlashLink technologies such as intelligent stripe aggregation and I/O priority scheduling, as well as multi-level caches, big data passthrough, and other key technologies to deliver high bandwidth and low latency. OceanStor Pacific series is adaptable to any customer need for I/O, latency, bandwidth, or capacity and supercharges the technology of today for the business of tomorrow.

1.2.2 Interworking of Unstructured Data Storage Services for On-Demand Use

OceanStor Pacific series supports the interworking of file, object, and HDFS unstructured data storage services. One piece of data can be shared and accessed by all unstructured services. The OceanStor Pacific series block service with an open architecture works seamlessly with a variety of containers and computing virtualization platforms to provide a data storage layer to various cloud infrastructures with scale-out capabilities on demand. This allows you to effortlessly build an open cloud platform without worrying over vendor lock-in when selecting infrastructures.

- Unstructured data storage services (unstructured services for short): provide the file service based on the NFS protocol, SMB protocol, standard POSIX interface, and MPI-IO library, meeting customers' requirements on file systems with high bandwidth and low latency in HPC scenarios. The unstructured services also provide the object service compatible with the Amazon S3 protocol and the HDFS service. Multiple unstructured services support

protocol interworking and mutual data access, eliminating data migration and saving space.

- Structured data storage service (structured service for short): provides standard access interface protocols such as SCSI and iSCSI, supports a wide range of virtualization platforms and database applications, and delivers superb performance and high scalability to meet SAN storage requirements of virtualization platforms, cloud resource pools, and databases.

1.2.3 Superb Capacity and Performance Density Design

OceanStor Pacific series leverages high-density hardware to help customers save equipment room space. Specifically, the ultra-large capacity design helps store more data with fewer devices; the superb performance density design helps quickly adapt to the all-flash trend.

OceanStor Pacific 9550 adopts the superb capacity density design where a high-cohesion structure is used to maximize the disk density. In addition, dual air channels and counter-rotating pressurized fans are used to resolve the heat dissipation problem. It is the first ultra-high-density storage device that accommodates to 1.1 m cabinets in the industry.

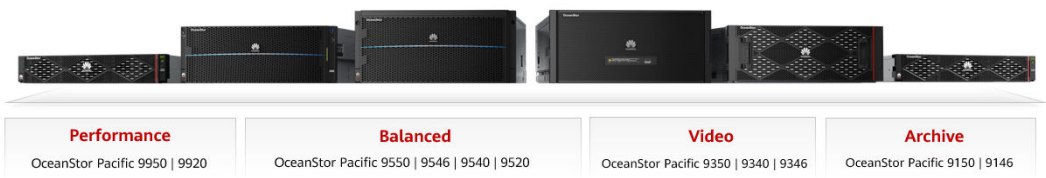
OceanStor Pacific 9950 adopts the superb all-flash performance density design. It is the first all-PCIe 4.0 flash storage device in the industry. It adopts the small-node design and uses Huawei-developed half-palm NVMe SSDs to provide customers with superb performance experience.

2 Hardware Architecture

- 2.1 Hardware Introduction
- 2.2 High-Performance Hardware Design
- 2.3 Large-Capacity Hardware Design

2.1 Hardware Introduction

Figure 2-1 Huawei OceanStor Pacific series storage system



OceanStor Pacific series provides end-to-end solution compatibility verification, covering various exception scenarios. The recommended typical configurations of storage devices are as follows:

Table 2-1 Typical configurations of OceanStor Pacific series storage devices

Category	Node Model	Configuration	Applicable Service			
			Block	File	Object	HDFS
Performance	OceanStor Pacific 9950	5 U 80-slot	-	√	√	√
	OceanStor Pacific 9920	2 U 25-slot	√	√	√	√
Balanced	OceanStor Pacific 9540	4 U 36-slot	√	√	√	√

	OceanStor Pacific 9520	2 U 12-slot	√	√	√	√
	OceanStor Pacific 9520	2 U 14-slot	√	√	√	√
	OceanStor Pacific 9520	2 U 16-slot	√	-	-	-
	OceanStor Pacific 9546	4 U 60-slot	-	√	√	√
	OceanStor Pacific 9550	5 U 120-slot	-	√	√	√
Video	OceanStor Pacific 9350	5 U 120-slot	-	√	√	-
	OceanStor Pacific 9346	4 U 60-slot	-	√	√	-
	OceanStor Pacific 9340	4 U 36-slot	-	√	√	-
Archive	OceanStor Pacific 9146	4 U 60-slot	-	√	√	-
	OceanStor Pacific 9150	5 U 120-slot	-	√	√	-

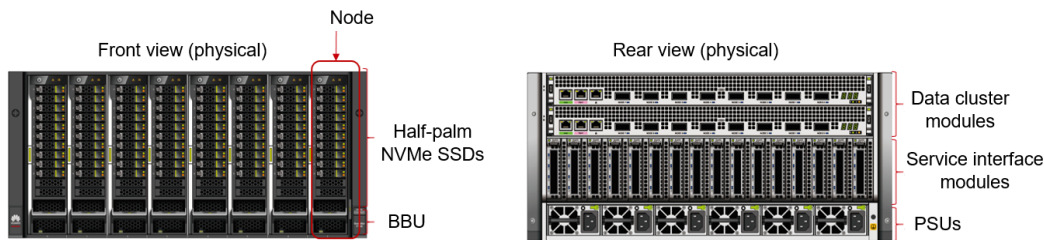
2.1.1 Dedicated Storage Hardware

2.1.1.1 OceanStor Pacific 9950

OceanStor Pacific 9950 is a 5 U high-density all-flash storage device. It adopts the multi-node design to deliver excellent storage performance.

- Each chassis of OceanStor Pacific 9950 houses 8 independent storage nodes, 80 half-palm NVMe SSD slots, 16 PCIe 4.0 service interface modules, and 16 x 100GE data cluster interfaces.
- Each node of OceanStor Pacific 9950 has 8 DDR4 memory channels. The node adopts the all-PCIe 4.0 design and can house 10 half-palm NVMe SSDs.

Figure 2-2 Huawei OceanStor Pacific 9950



2.1.1.2 OceanStor Pacific 9546

OceanStor Pacific 9546 is a 4 U high-density storage device developed based on Huawei Kunpeng 920 processors. It delivers superb reliability and disk density.

- Each chassis of OceanStor Pacific 9546 provides 60 disk slots, a built-in guide rail and a cable tray, and full-FRU design. It supports cabinets with a depth of 1000 mm or above.
- OceanStor Pacific 9546 has two configurations: one node per chassis and two nodes per chassis.
- For the one node per chassis configuration of OceanStor Pacific 9546, each node uses two 48-core Kunpeng processors and 16 DDR4 memory channels. For the two nodes per chassis configuration, each node uses one 64-core Kunpeng processor or two 48-core Kunpeng processors and 8 or 16 DDR4 memory channels. Each node adopts the all-PCIe 4.0 design.

Figure 2-3 Front panel of OceanStor Pacific 9546

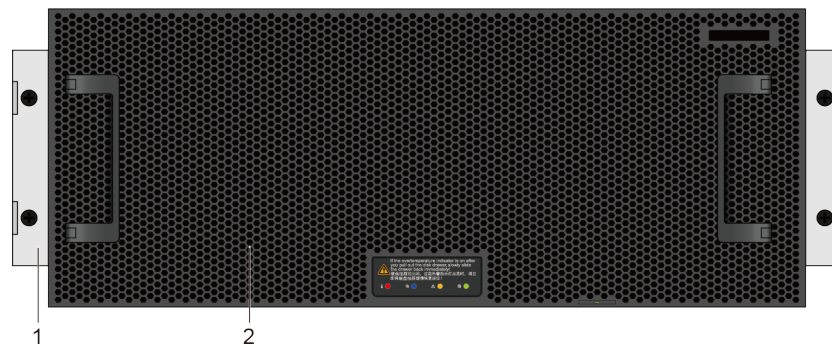


Figure 2-4 Rear panel of OceanStor Pacific 9546 (One Node per Chassis)

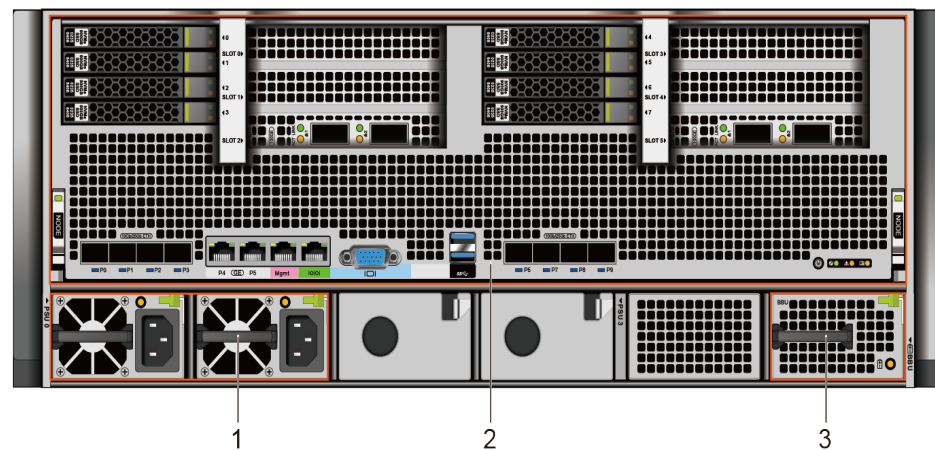
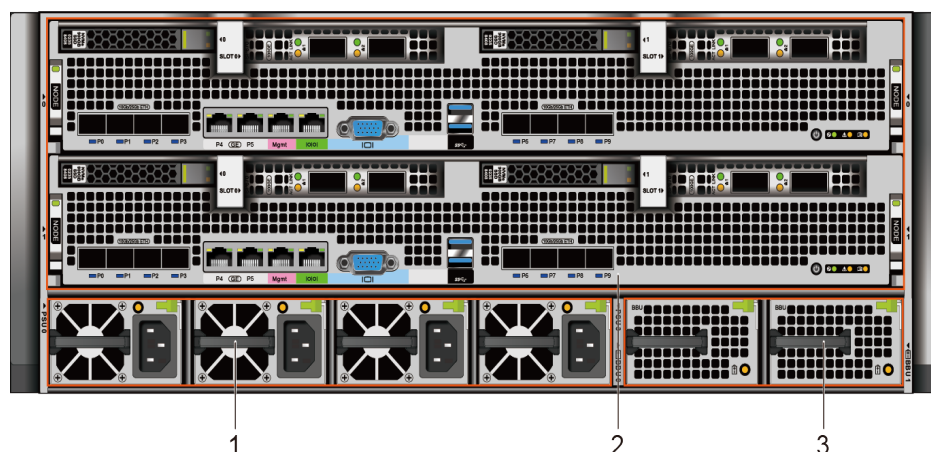


Figure 2-5 Rear panel of OceanStor Pacific 9546 (Two Nodes per Chassis)

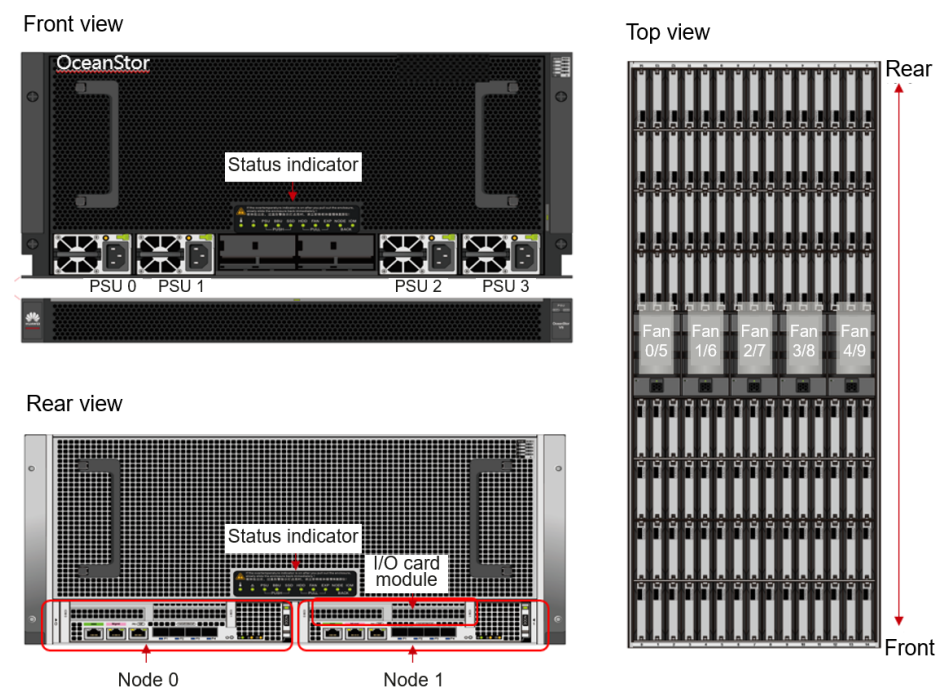


2.1.1.3 OceanStor Pacific 9550

OceanStor Pacific 9550 is a 5 U high-density and large-capacity storage device. It adopts dedicated two-node scale-out hardware design to deliver superb reliability and disk density.

- Each chassis of OceanStor Pacific 9550 houses 2 nodes and 120 disk slots, providing the highest density (24 disks per U) in the industry.
- Each node of OceanStor Pacific 9550 delivers a 20 W energy-efficient access capability per disk. It is equipped with built-in guide rails and tank chains, and adopts the full-FRU design.

Figure 2-6 Huawei OceanStor Pacific 9550

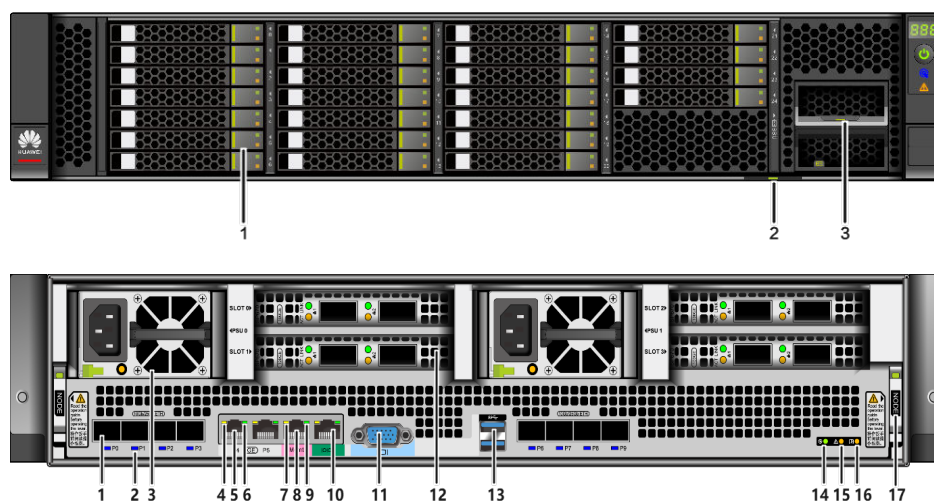


2.1.1.4 OceanStor Pacific 9920

OceanStor Pacific 9920 is a 2 U high-density all-flash storage device. It uses the latest NVMe SSDs as main storage disks to provide ultimate storage performance for customers.

- An OceanStor Pacific 9920 storage device supports 25 palm-sized NVMe SSD slots and 2 or 3 PCIe 4.0 service interface modules.
- Each OceanStor Pacific 9920 node provides 16 DDR4 memory channels

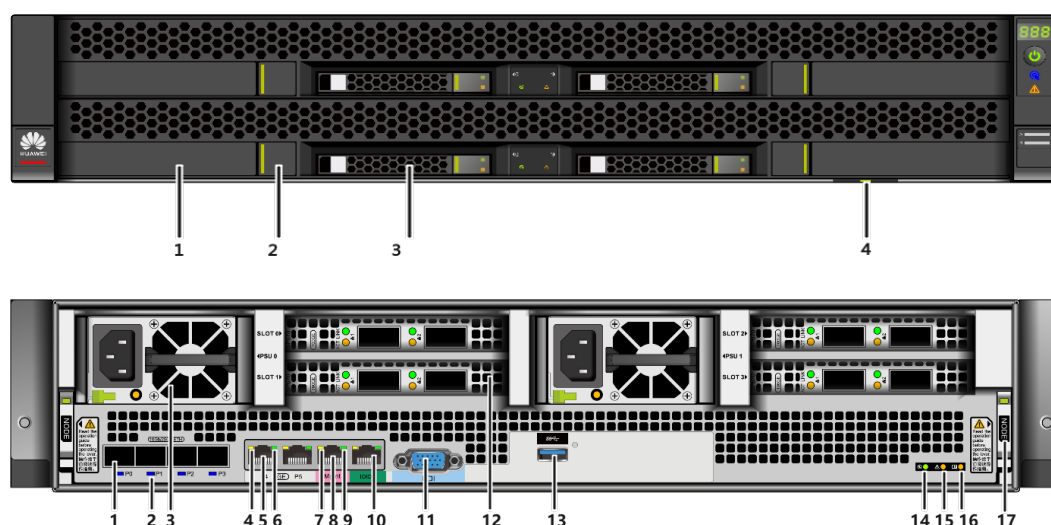
Figure 2-7 Huawei OceanStor Pacific 9920



2.1.1.5 OceanStor Pacific 9520

OceanStor Pacific 9520 14-slot or 16-slot node is a dedicated storage node developed based on Huawei Kunpeng 920 processor. It features low power consumption, simplified management, and easy deployment.

Figure 2-8 Huawei OceanStor Pacific 9520

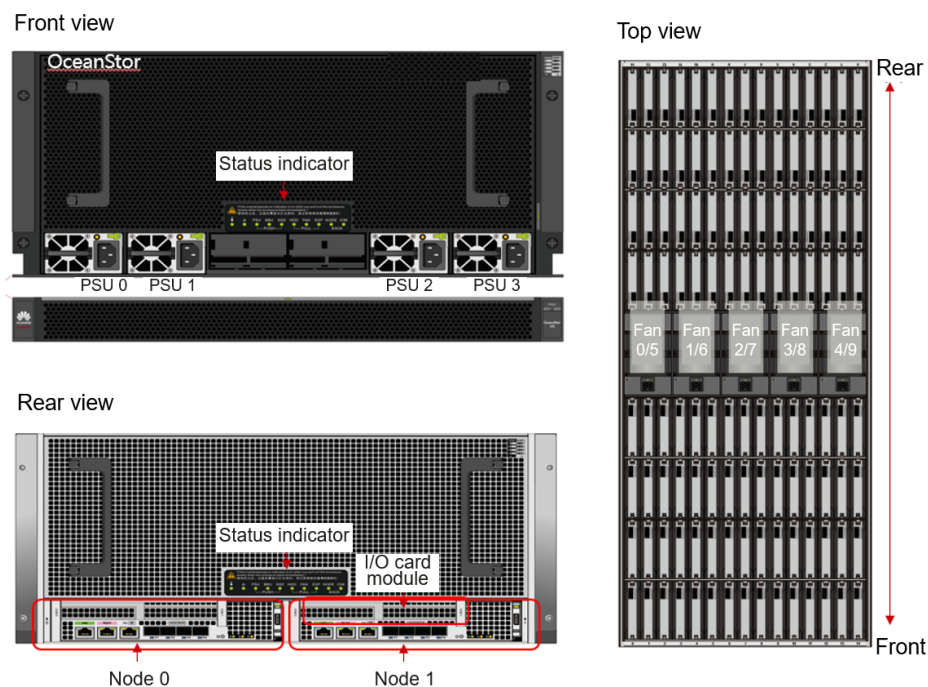


2.1.1.6 OceanStor Pacific 9350

OceanStor Pacific 9350 is a 5 U high-density and large-capacity storage device. It adopts dedicated two-node scale-out hardware design to deliver superb reliability and disk density. The device is mainly used in intelligent video and image scenarios.

- Each chassis of OceanStor Pacific 9350 houses 2 nodes and 120 disk slots, providing the highest density (24 disks per U) in the industry.
- Each node of OceanStor Pacific 9350 delivers a 20 W energy-efficient access capability per disk. It is equipped with built-in guide rails and tank chains, and adopts the full-FRU design.

Figure 2-9 Huawei OceanStor Pacific 9350



2.1.1.7 OceanStor Pacific 9346

OceanStor Pacific 9346 is a 4 U high-density storage device developed based on Huawei Kunpeng 920 processors. It delivers superb reliability and disk density. The device is mainly used in intelligent video and image scenarios.

- Each chassis of OceanStor Pacific 9346 provides 60 disk slots, a built-in guide rail and a cable tray, and full-FRU design. It supports cabinets with a depth of 1000 mm or above.
- OceanStor Pacific 9346 provides the following configuration: one node per chassis.
- Each node of OceanStor Pacific 9346 adopts 2 x 48-core Kunpeng processors, 16 x DDR4 memory channels, and all-PCIe 4.0 design.

Figure 2-10 Front panel of OceanStor Pacific 9346

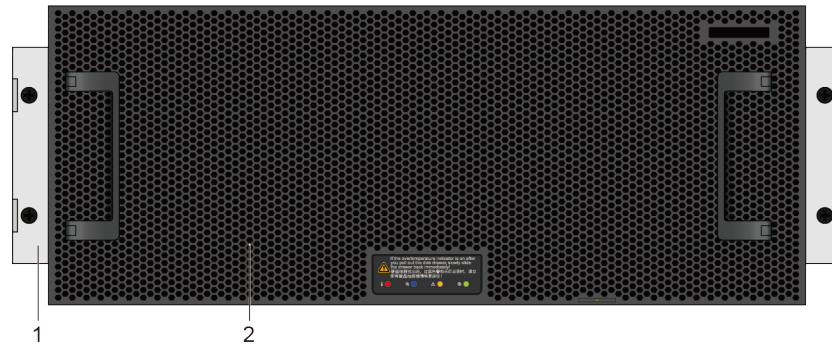
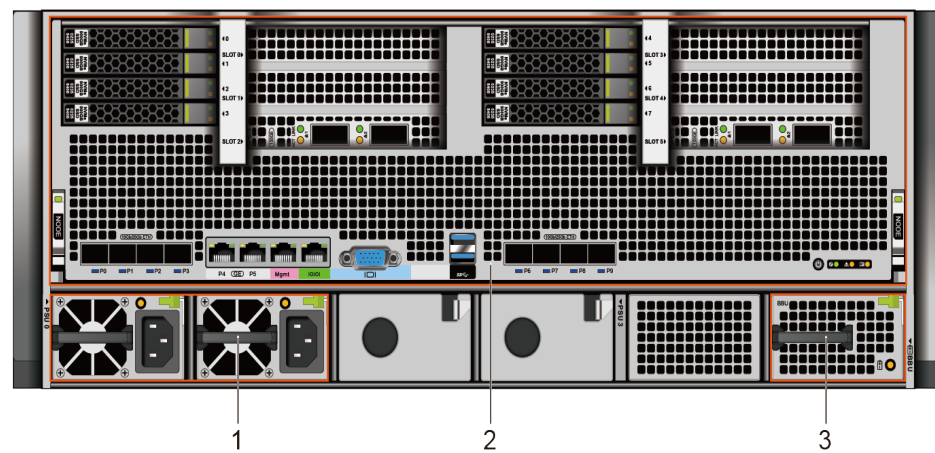


Figure 2-11 Rear panel of OceanStor Pacific 9346 (One Node per Chassis)

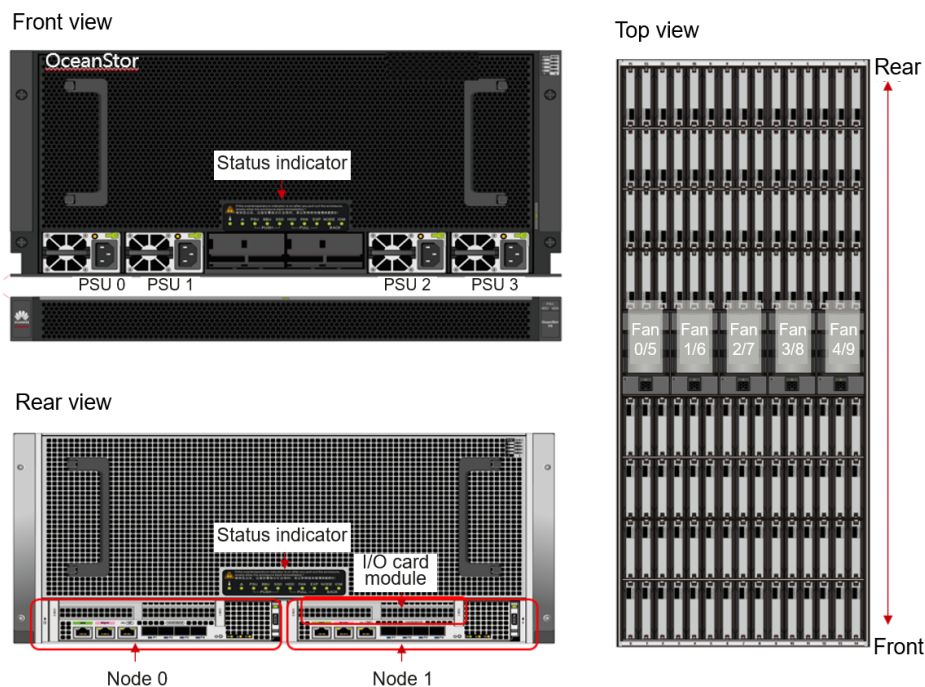


2.1.1.8 OceanStor Pacific 9150

OceanStor Pacific 9150 is a 5 U high-density and large-capacity device, and mainly used in backup and archiving scenarios. It adopts dedicated two-node scale-out hardware design and industry-leading capacity density to provide cost-effective data access services for long-term retention of mass data.

- Each chassis of OceanStor Pacific 9150 houses 2 nodes and 120 disk slots, providing the highest density (24 disks per U) in the industry.
- Each node of OceanStor Pacific 9150 delivers a 20 W energy-efficient access capability per disk. It is equipped with built-in guide rails and tank chains, and adopts the full-FRU design.

Figure 2-12 Huawei OceanStor Pacific 9150



2.1.1.9 OceanStor Pacific 9146

OceanStor Pacific 9146 is a 4 U high-density storage device developed based on Huawei Kunpeng 920 processors. It is a brand-new storage option that balances capacity, performance, and costs, and provides cost-effective data access services for long-term retention of mass data.

- Each chassis of OceanStor Pacific 9146 provides 60 disk slots, a built-in guide rail and a cable tray, and full-FRU design. It supports cabinets with a depth of 1000 mm or above.
- OceanStor Pacific 9146 provides the following configuration: one node per chassis.
- Each node of OceanStor Pacific 9146 adopts one 64-core Kunpeng processor, eight DDR4 memory channels, and all-PCIe 4.0 design.

Figure 2-13 Front panel of OceanStor Pacific 9146

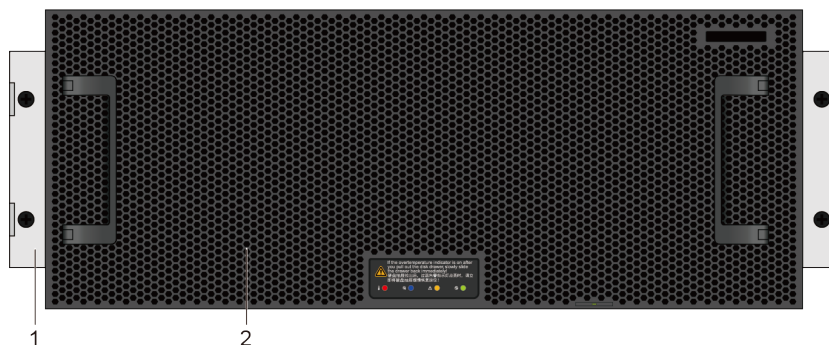
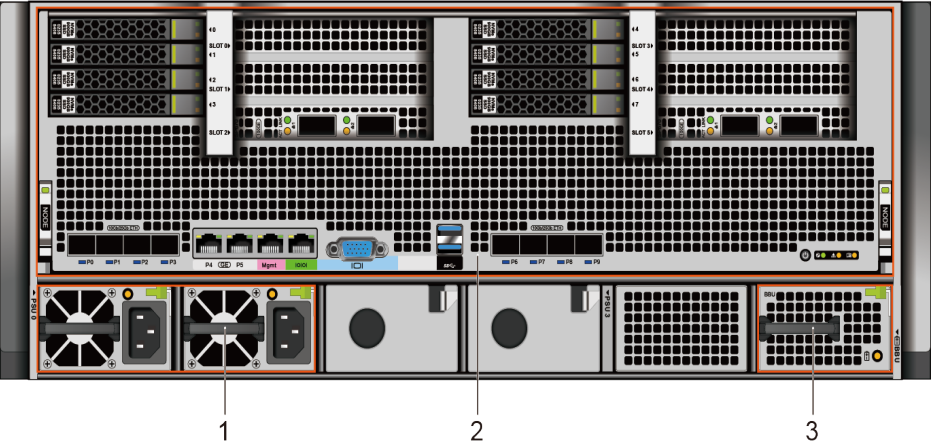


Figure 2-14 Rear panel of OceanStor Pacific 9146 (One Node per Chassis)



2.1.2 Universal Storage Hardware

2.1.2.1 OceanStor Pacific 9540

OceanStor Pacific 9540 is 4 U device with 36 disk slots.

Figure 2-15 Front panel of OceanStor Pacific 9540

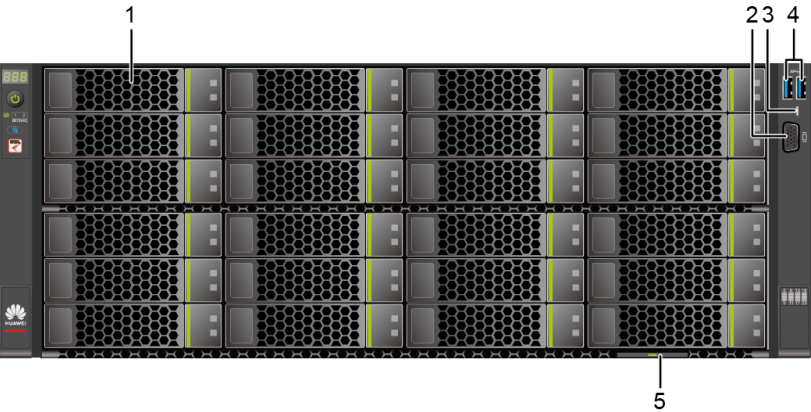
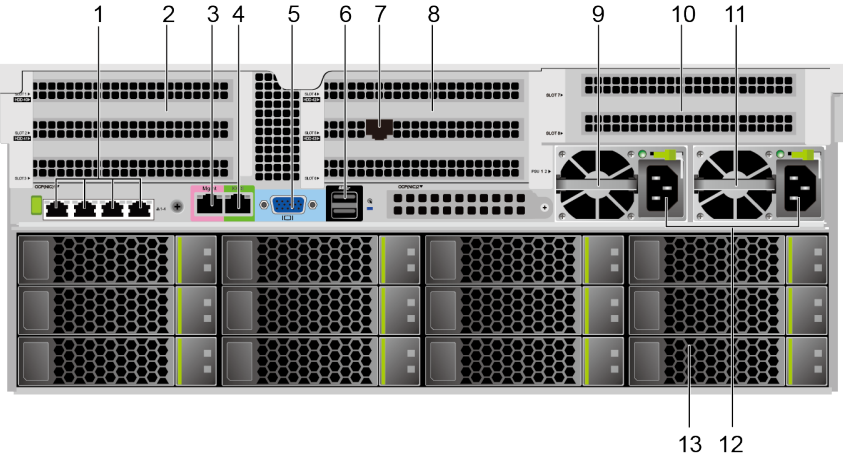


Figure 2-16 Rear panel of OceanStor Pacific 9540



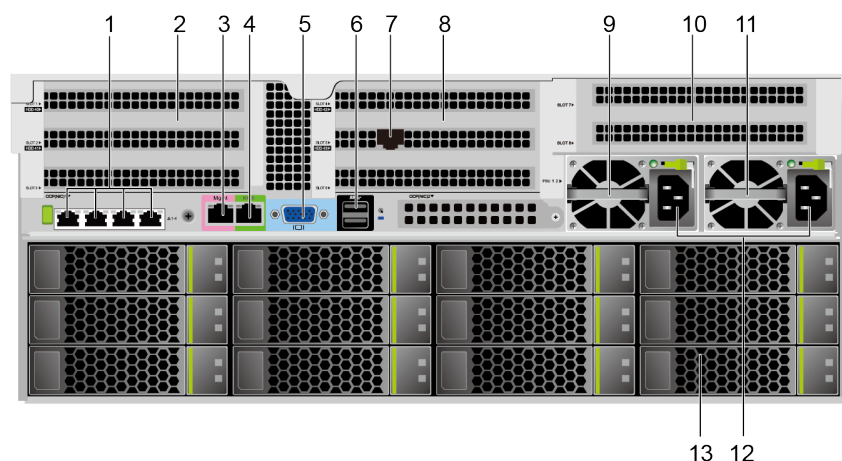
2.1.2.2 OceanStor Pacific 9340

OceanStor Pacific 9340 is a 4 U dual-socket rack capacity-intensive storage node developed based on Huawei Kunpeng 920 processors. The device provides 36 disk slots and is mainly used in intelligent video and image scenarios.

Figure 2-17 Front panel of OceanStor Pacific 9340



Figure 2-18 Rear panel of OceanStor Pacific 9340



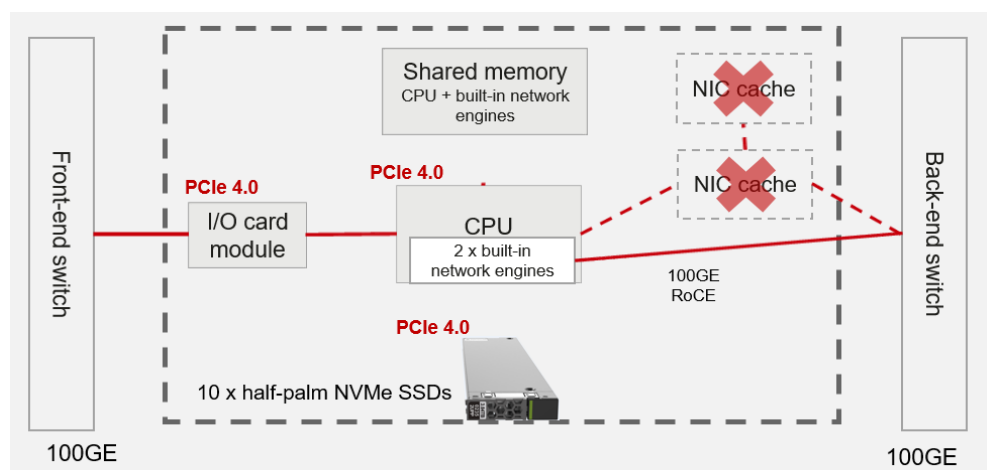
2.2 High-Performance Hardware Design

OceanStor Pacific 9950 adopts the superb all-flash performance density design and the small-node design. A single chassis is 5 U high and can house 8 nodes with 80 disks in total. A single device delivers up to 160 GB/s performance.

2.2.1 Front- and Back-End PCIe 4.0

The entire system adopts the PCIe 4.0 design. The industry's first all-PCIe 4.0 all-flash system doubles the bandwidth compared with the existing PCIe 3.0 system, providing higher bandwidth and lower latency.

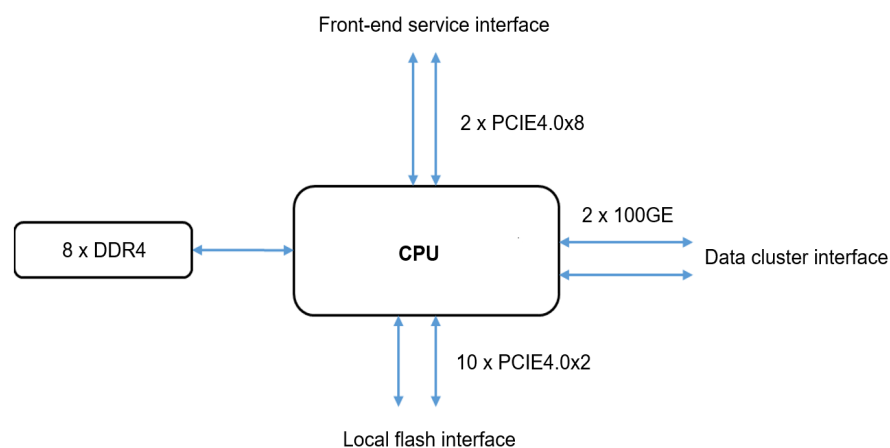
Figure 2-19 Front- and back-end PCIe 4.0 of Huawei OceanStor Pacific 9950



Two PCIe4.0X8 interfaces are allocated for the front-end service interface bandwidth of the CPU, providing a total bandwidth of 20 GB/s. OceanStor Pacific 9950 uses the back-end NIC-free design to maximize the access capability of the CPU and reduce costs. In addition, the back-end data cluster uses two 100GE ports provided by the CPU. The ports support the RDMA technology, greatly reducing the network latency, effectively transferring cluster data to other nodes, and consuming only a small amount of CPU resources. The local flash memory interface uses 10 PCIe4.0X2 interfaces and the SSD interface protocol is NVMe, which enables the CPU to access the media through PCIe. Compared with the traditional SAS protocol, the I/O latency can be reduced by more than 50%.

Each node uses the optimal bandwidth ratio to achieve the optimal bandwidth allocation relationship. Front-end:cluster:media = 1:1:1.25 (bandwidth ratio), fully utilizes the CPU bandwidth capability.

Figure 2-20 Channel bandwidth ratio of Huawei OceanStor Pacific 9950

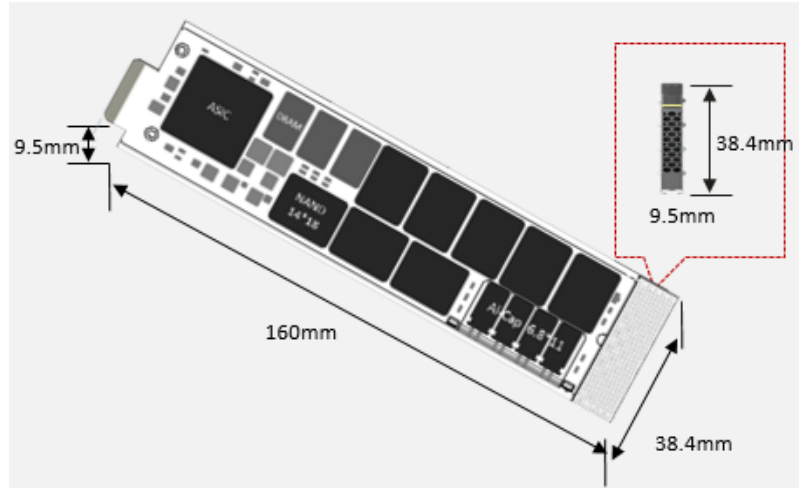


2.2.2 Huawei-Developed Half-Palm NVMe SSD

OceanStor Pacific 9950 uses Huawei-developed NVMe SSDs and supports 80 half-palm NVMe SSDs. This media form is specially designed for flash storage. One half-palm NVMe SSD supports a capacity of up to 7.68 TB. Compared with

traditional SSDs, half-palm NVMe SSDs are lighter and thinner, occupy less space, and deliver superior performance. Compared with traditional NVMe SSDs, more half-palm NVMe SSDs can be placed on each rack, releasing more rack space with the same capacity, reduces equipment room costs.

Figure 2-21 Huawei-developed half-palm NVMe SSD

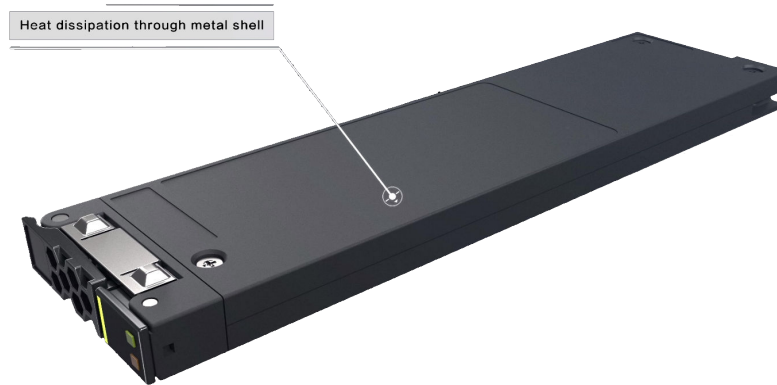


To maximize SSD performance, OceanStor Pacific 9950 uses the Kunpeng multi-core processors to provide powerful computing capabilities. The micro-architecture designs, such as branch prediction algorithm optimization, computing unit quantity increase, and memory subsystem architecture improvement, greatly improve processor performance and lay a hardware foundation for the CPU group-based and core-based algorithm to optimize system latency. The Kunpeng processor integrates three types of chips: southbridge, NIC, and SAS storage node, achieving industry-leading integration. A single chip implements the functions of four traditional chips and supports dynamic frequency modulation, reducing the TCO.

2.2.3 Efficient Heat Dissipation Design

The power consumption of the flash media increases as the performance increases. How to address the heat dissipation problem is also a difficulty. In OceanStor Pacific 9950, a total of 32 counter-rotating fans with a diameter of 40 mm (the counter-rotating fans are located in a node and can be replaced independently after the node is removed) are used to provide sufficient air volume for half-palm NVMe SSDs, CPUs, memory modules, and I/O card modules and the heat is efficiently brought into the air. The air conditioners in the equipment room take the heat out of the equipment room. The parallel backplane design is used to connect SSDs and CPUs. The advantage of the parallel backplane is that when an SSD is connected to a mainboard, the air flow direction is not blocked by the vertical backplane. In addition, the backplane uses an ultimate window opening design. The minimum wind resistance effectively increases the air volume for heat dissipation of SSDs and reduces the SSD temperature. After counter-rotating fans are used, the operating temperature of disks can be lower than 35°C at a normal ambient temperature, creating a suitable operating environment for disks.

Figure 2-22 Heat dissipation shell of Huawei-developed half-palm NVMe SSD



A half-palm NVMe SSD has a size of 160 mm x 38.4 mm x 9.5 mm and uses an all-metal shell design. The shell material is full die-casting aluminum, and the thermal conductivity is higher than 200 W/mK. The internal particles and chips can contact the shell, effectively transferring the heat inside the disk, reducing the operating temperature of flash particles by 5°C compared with other materials. To ensure the reliable operation of the processor, the VC phase-change radiator and carbon fiber thermal pad are used. This effectively ensures that the junction temperature of the CPU is lower than 85°C, which not only ensures the secure and reliable operation of the CPU, but also maximizes the performance of the CPU.

2.3 Large-Capacity Hardware Design

The 4 U space of a common scale-out storage node houses a maximum of 36 x 3.5-inch disks. OceanStor Pacific series can house 120 x 3.5-inch disks in a 5 U space. One device is equivalent to three 36-disk universal nodes. Its storage density is 2.67 times that of the latter. The benefit brought by the storage density improvement is space saving. Compared with a 36-disk node, the space occupied by the same number of disks is reduced by 62.5%.

The high-density hardware design needs to resolve the following problems:

- To ensure the heat dissipation efficiency of disks, sufficient space is required between disks.
- The space loss of disk enclosures is reduced and more space is reserved for disks.
- All disks can be maintained independently. This requires that the entire disk enclosure can be pulled out of a cabinet. To support the pulling of a disk enclosure with a weight of more than 100 kg and prevent the cabinet from tilting, holding rails are commonly used in the industry, which occupies the disk assembly space.

OceanStor Pacific 9550 uses the bi-directional double-layer drawer, tank chain, and retraction designs to maximize the disk density. This resolves the conflict between the ultimate density and O&M space, and resolves the disk heat dissipation and online independent maintenance problems.

2.3.1 High-Cohesion Structure Design

The bi-directional double-layer drawer design resolves the following problems: maintenance of upper-layer disks and separation of lower-layer nodes and the

fixed parts such as PSUs and FRUs. The two layers are connected through a movable tank chain. OceanStor Pacific 9550 adopts a retraction design for nodes and PSUs in the lower 1 U space, saving the cabling space of the nodes and PSUs. This is the industry's first ultra-high-density device that meets the requirements of 1.1 m deep universal cabinets.

Table 2-2 Bi-directional double-layer drawer of Huawei OceanStor Pacific 9550



You do not need to open the upper cover of the chassis to maintain all disks. You can pull out 60 disks from the front or rear of the chassis for maintenance. The tank chain design ensures that services of 120 disks are not interrupted during operation and maintenance. When a disk is dragged in a 4 U chassis, one end of a high-speed signal cable is connected to the 4 U disk chassis and the other end is connected to the node backplane, and all cables are bound in the tank chain and move with the chassis.

Figure 2-23 Chassis structure of Huawei OceanStor Pacific 9550



Bi-directional pulling ensures that the center of gravity of the chassis is in the chassis. In this way, holding rails are not required, greatly reducing the width of sliding rails and leaving more space for disks and heat dissipation channels between disks. In this way, the distance between disks is increased by 50% and the heat dissipation capability is improved by 30%, supporting the stable running of 120 disks. Because the center of gravity of the chassis is in the chassis, the center of gravity deviation is reduced by 75%, which ensures the security when the device is installed at a high position in the cabinet.

2.3.2 Efficient Heat Dissipation Design

Figure 2-24 Heat dissipation technology of Huawei OceanStor Pacific 9550



After the disk density problem is resolved, the components that generate much heat, such as CPUs and memory modules, in the two nodes of the lower 1 U space cannot be ignored. If the heat dissipation problem cannot be effectively resolved, the failure rate will be high and the system may even break down. To ensure the heat dissipation of the CPU, OceanStor Pacific 9550 uses a carbon fiber thermally conductive film + vacuum VC radiator technology. The carbon fiber thermally conductive film has an ultra-high thermal conductivity coefficient, light weight, and corrosion resistance, and is very suitable for a device with a high density and a high dimension precision requirement. The vacuum VC evenly distributes a heat flow dissipated by a plurality of heat sources to a relatively large heat dissipation area within a short distance. Compared with linear heat conduction of the copper conduit, the vacuum VC has a higher heat dissipation rate. With this technology, the overall heat dissipation capability is improved by 20%. Compared with the traditional heat dissipation technology, the CPU core temperature is reduced by 11°C.

In addition, fans are installed in the middle and two layers of independent air channel are used in OceanStor Pacific 9550. The air passes 60 disks in the front and then 60 disks in the rear, maximizing the capability of the axial fans. Nodes, PSUs, batteries, and SSD caches are located in the lower part of the chassis, forming two layers of independent air channels. This prevents disks, CPUs, and memory modules from affecting each other in a high-temperature environment. Compared with the heat dissipation technology of general-purpose servers, the component failure rate is reduced by 50%.

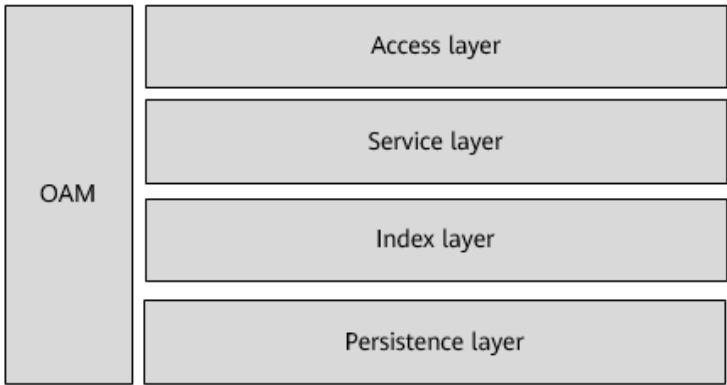
3 Software Architecture

- 3.1 Fully Symmetric Scale-out Software Architecture
- 3.2 Global Load Balancing
- 3.3 Multi-Level Cache Acceleration
- 3.4 Key I/O Processes
- 3.5 Convergence and Interworking of Unstructured Data Storage Services

3.1 Fully Symmetric Scale-out Software Architecture

3.1.1 Logical Software Architecture

Figure 3-1 Logical software architecture of OceanStor Pacific series products

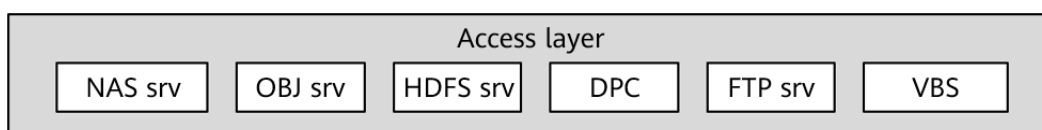


OceanStor Pacific series is an intelligent scale-out storage product that supports mass scale-out and provides block, file, object, and HDFS services. Users can deploy storage services as required. The global namespace provides a unified storage pool where storage space can be centrally managed and allocated. Operating the global namespace avoids maintaining multiple namespaces during capacity expansion. As shown in the preceding figure, Operation Administrator

and Maintenance (OAM) provides management and control services of OceanStor Pacific series, including resource management, service management, system management, user management, installation and deployment, upgrade, capacity expansion and reduction, as well as inspection and information collection. The I/O plane is logically divided into the access layer, service layer, index layer, and persistence layer from top to bottom.

3.1.1.1 Access Layer

Figure 3-2 Functional elements of the access layer



Functionally, the access layer receives and processes the access requests of various services. For unstructured services, the access layer consists of the following functional modules:

- The NAS srv module provides a standard NAS protocol (NFS or SMB) for access processing.
- The OBJ srv module provides a standard S3 protocol for object access processing.
- The HDFS srv module provides a standard HDFS protocol for access processing.
- The Distributed Parallel Client (DPC) module is designed for high-performance computing (HPC) scenarios and is compatible with standard POSIX/MPI-IO semantic access.
- The FTP srv module provides a standard FTP/FTPS protocol for access processing.

For the structured service, Virtual Block Service (VBS) provides standard iSCSI access and provides local SCSI semantic access for high-performance scenarios.

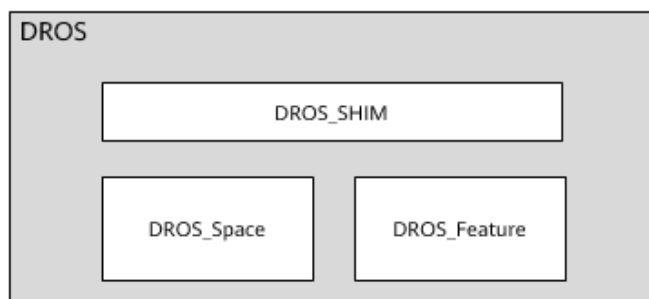
3.1.1.2 Service Layer

Figure 3-3 Functional elements of the service layer



The service layer provides structured and unstructured services. All unstructured services are processed on the Distributed Relational Object Service (DROS) platform in a unified manner and they share the same metadata, data, and value-added features. The structured service is processed by an independent block module.

Figure 3-4 Functional elements of DROS



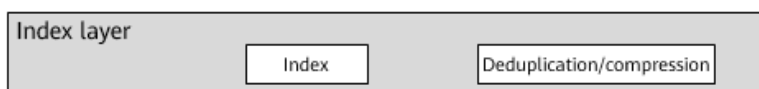
In the DROS module, DROS_SHIM is the semantic abstraction layer, which abstracts the semantics of unstructured services including file, HDFS, and object in a unified manner. After different upper-layer unstructured semantics are accessed, the corresponding semantic operations are mapped to the function interfaces provided by DROS_SHIM.

DROS_Space provides metadata processing for basic unstructured services to ensure that all unstructured services share the same metadata.

DROS_Feature provides unified value-added services for unstructured services, such as quota, tiering, and snapshot.

3.1.1.3 Index Layer

Figure 3-5 Functional elements of the index layer



The index layer provides fine-grained space layout management capabilities above the persistence layer.

The index module maps the data objects in the DROS or block module to the lower-layer Plog space, and provides cache capabilities based on battery-protected memory for unstructured services.

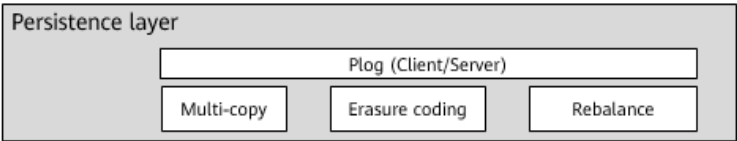
The deduplication/compression module provides deduplication and compression capabilities for the structured service to reduce costs.

NOTE

A persistence log (Plog) is a set of physical addresses managed in a fixed size at the persistence layer. It provides EC or multi-copy redundancy protection. The upper layer service accesses the LBA address on a specific physical disk in Plog ID+offset mode.

3.1.1.4 Persistence Layer

Figure 3-6 Functional elements of the persistence layer



The persistence layer allocates storage space and handles faults on storage media such as HDDs and SSDs, and provides reliable and scale-out data management through cross-node EC or copies. The persistence layer focuses on underlying storage capabilities such as reliability, performance, and scalability, and provides Plog semantics-based space access capabilities for the upper layer.

3.1.1.5 Microservice Forms

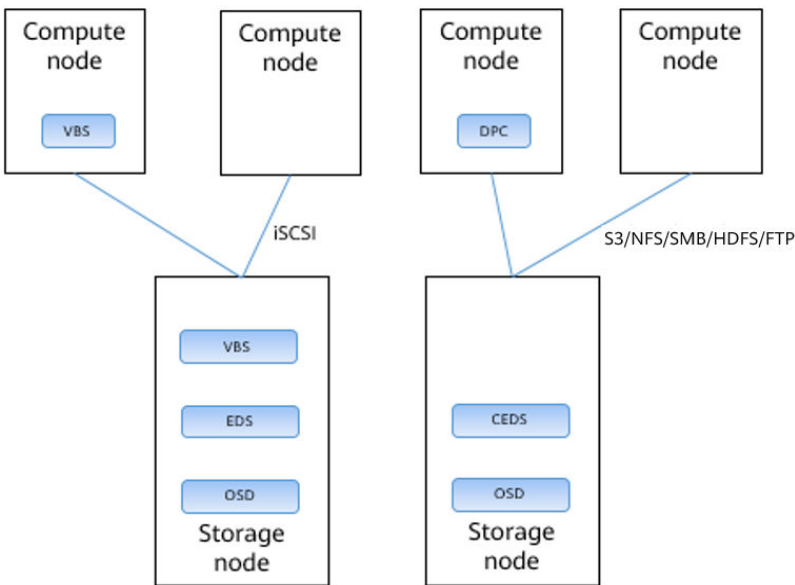
The logical software architecture of OceanStor Pacific series products is described above. The entire product software is released in the form of microservices. The preceding software architecture elements are built in the corresponding microservice packages. The key microservice components are as follows:

Table 3-1 Microservice component list

Microservice Package Name	Function	Logical Function Architecture Elements
DPC	Deployed on compute nodes and provides standard POSIX and MPI-IO access.	DPC, PlogClient
VBS	Provides standard iSCSI/SCSI access and can be deployed on compute nodes or storage nodes.	VBS
Enterprise Data Service (EDS)	Provides the structured service and value-added features.	Block, index layer, and PlogClient
Converged Enterprise Data Service (CEDS)	Provides converged unstructured services.	NAS srv, OBJ srv, HDFS srv, FTP srv, DROS, index layer, and PlogClient
Object Storage Device (OSD)	Provides data persistence capabilities.	Persistence layer

The following figure shows the deployment of these key microservice components.

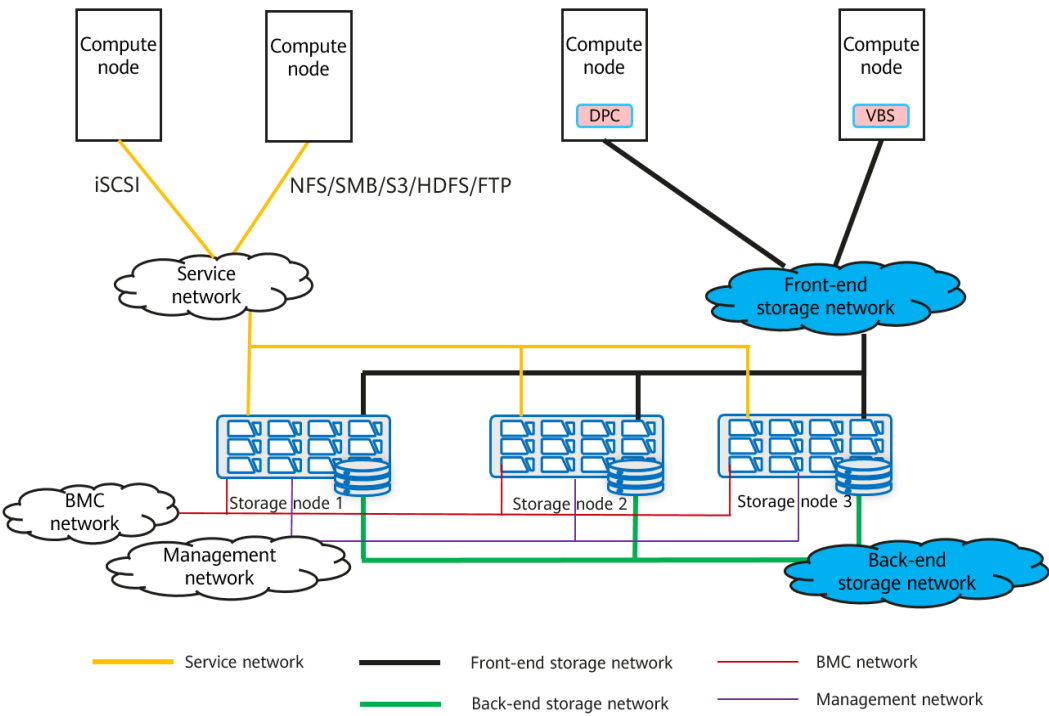
Figure 3-7 Deployment of key microservice components in OceanStor Pacific series



The VBS microservice component can be deployed on storage nodes to provide iSCSI access for external systems. The VBS microservice component can also be deployed on compute nodes to provide higher-performance access. The DPC microservice component is deployed on compute nodes to provide higher performance in HPC scenarios.

3.1.2 Logical Network Architecture

Figure 3-8 Logical network design of OceanStor Pacific series



Based on factors such as maintenance and management, performance, and reliability, OceanStor Pacific series classifies and defines network types, including the service network, front-end storage network, back-end storage network, management network, and BMC network from user application access to storage node scale-out.

Table 3-2 Logical network function description

Name	Description	Network Type	Service Scenario
Storage service network	Used for client access over standard protocols.	1. TCP/IP protocol 2. IB protocol 3. RoCE protocol	1. Used for compute node access over the NFS/SMB/S3/HDFS/FTP/FTPS protocol. 2. NFS over RDMA uses the IB or RoCE network. 3. If VBS is not deployed on compute nodes, iSCSI is used for access.

Name	Description	Network Type	Service Scenario
Front-end storage network	Used for the data communication between DPC/VBS and storage nodes when DPC/VBS is deployed on compute nodes.	1. TCP/IP protocol 2. IB protocol 3. RoCE protocol	1. In scenarios that do not require high performance, the front- and back-end storage networks can be integrated, which is recommended in standard protocol access scenarios.
Back-end storage network	Data communication network between storage nodes. It is used to rebuild and balance background traffic to ensure that the host I/O performance is not affected by background I/Os. RDMA is supported.	1. TCP/IP protocol 2. IB protocol 3. RoCE protocol	2. In scenarios where DPC is used in the file service, it is recommended that the front- and back-end storage networks be deployed separately and RDMA be used to improve performance. 3. On a 100GB storage network, it is recommended that the NFS over RDMA service be deployed together with the storage network. If the storage network bandwidth is less than 100GB, you are advised to use an independent storage service network.
Management network	Used for storage system management and maintenance.	TCP/IP protocol	One or two GE network ports are supported. Two network ports can be bonded in bond1, bond2, or bond4 mode and bond1 is recommended.
BMC network	Provides remote out-of-band hardware device management.	TCP/IP protocol	GE network ports for remote hardware device management

3.2 Global Load Balancing

OceanStor Pacific series in fully symmetric scale-out software architecture consists of two layers of routes. One is at the service layer to ensure that each node can share the service processing capabilities, and the other one is at the persistence layer to balance storage space and accelerate data rebuilding upon faults.

Unstructured and structured services at the service layer have different route-based hash policies due to semantic differences. However, both unstructured and structured services are mapped based on the same basic routing model. The route-based hash policy of the persistence layer is based on STORE DHT. The following sections describe front-end load balancing and back-end load balancing.

3.2.1 Front-End Load Balancing

The front-end load balancing of unstructured and structured services in OceanStor Pacific series is based on the same basic routing model. The following describes the basic routing model first and then describes the differentiated parts. For unstructured services, due to protocol interworking, the front-end access model is abstracted and normalized (see [3.5 Convergence and Interworking of Unstructured Data Storage Services](#)), and this section mainly describes file semantics.

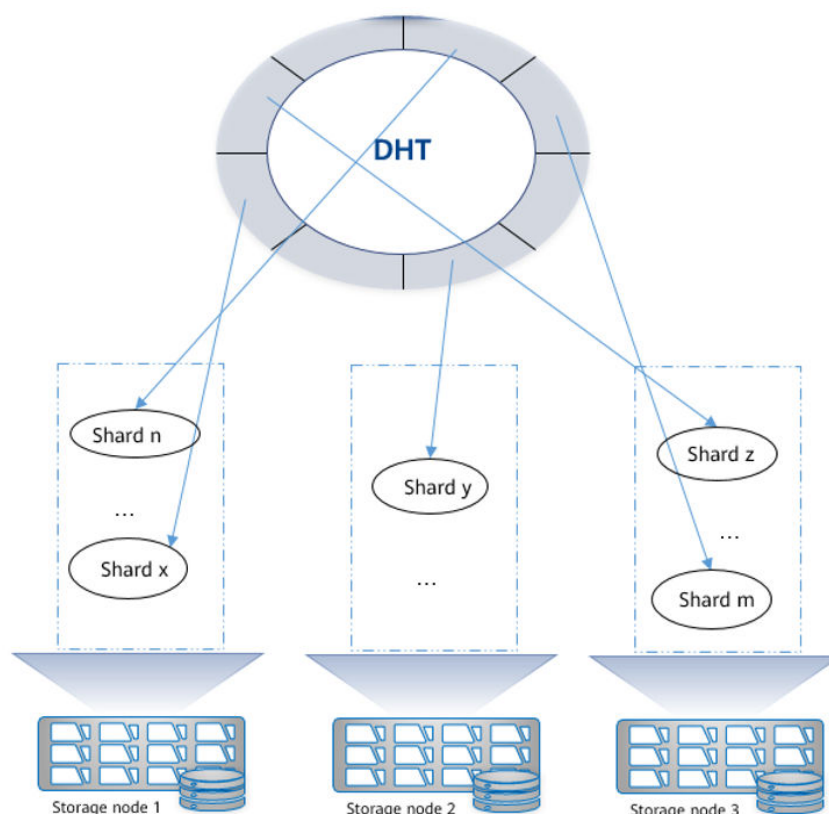
3.2.1.1 Basic Routing Model

OceanStor Pacific series storage adopts the shard DHT routing algorithm as the basic routing policy of the service layer. Each storage pool has a shard DHT hash ring. Each hash ring has 4096 shards, and the 4096 shards are evenly allocated to the nodes in a storage pool. Suppose that a storage pool has 16 nodes. The 4096 shards will be evenly allocated to the 16 nodes. That is, each node processes 256 shards (that is, $4096/16$).

The shard DHT routing algorithm adopted by OceanStor Pacific series has the following characteristics:

- **Balance:** Data is distributed to all nodes as evenly as possible, thereby balancing loads among nodes.
- **Monotonicity:** When new nodes are added to the system, the system redistributes data among nodes. The new nodes take over the data service corresponding to the shard processing unit. Data will be allocated to the new nodes for processing, and the data on the existing nodes is not significantly adjusted.

Figure 3-9 Shard DHT routing

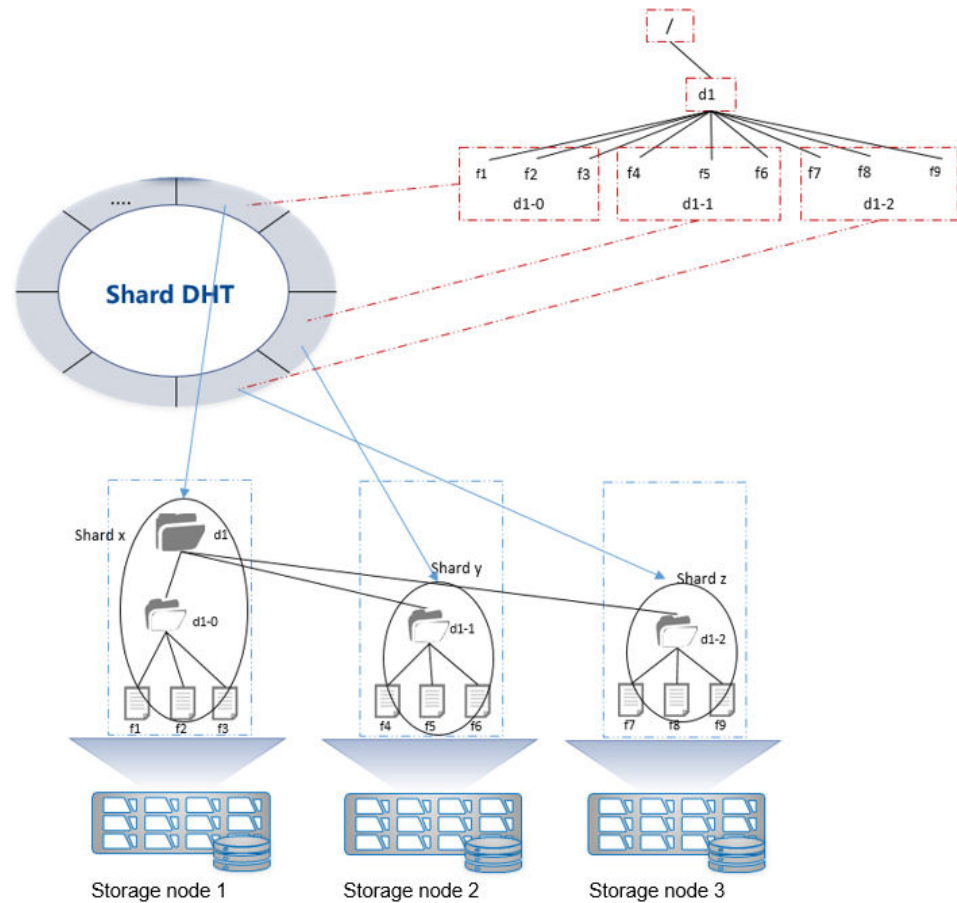


3.2.1.2 Route Mappings for File Semantics

Objects of the OceanStor Pacific series are mapped to files. Buckets of objects are mapped to root directories. Unstructured services share the same metadata management. Objects, HDFS, and file services use the same hash policy. Specifically, in order to solve the performance problem of excessive large files, a directory fragment-based distributed hash policy is adopted to achieve load balancing. A directory fragment and its sub-files belong to the same node for processing. The following uses a file as an example to describe the hash policy.

When creating a directory, the system creates multiple directory fragments by default and selects a shard for each directory fragment. When a file is created in this directory, the hash algorithm is used to calculate the directory fragment to which the file belongs. Related metadata operations are forwarded to the node to which the corresponding directory fragment belongs for processing.

Figure 3-10 Directory fragment-based routing

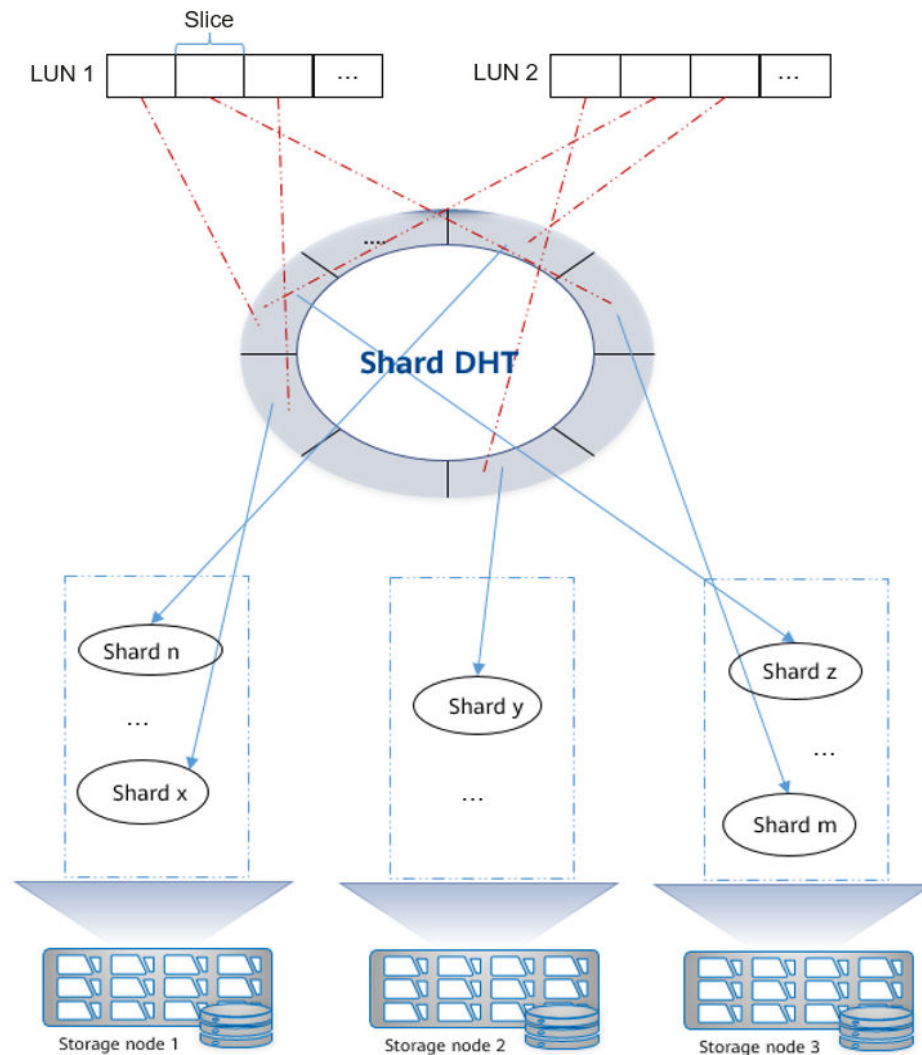


As shown in the preceding figure, when directory **d1** is created, the system creates three directory fragments **d1-0**, **d1-1**, and **d1-2** for it, selects three shards for the three directory fragments, and then determines the nodes to which each directory fragment belongs based on the mappings between the shards and storage nodes. The files created in directory **d1** will be hashed to a directory fragment based on the file name first. The corresponding file metadata operations will be forwarded to the node to which the corresponding directory fragment belongs for processing.

3.2.1.3 Route Mappings for Block Semantics

OceanStor Pacific series block service divides each LUN into multiple slices at a fixed granularity (for example, 4 MB) for LUN data balancing. Each slice is calculated using the hash factor of the LUN (that is, LUN ID) and start LBA of the shard to obtain the hash value of the slice. Then, the system locates each slice in the DHT ring shown in the following figure according to hash values.

Figure 3-11 Block semantic routing



Take the block service as an example:

When an upper-layer application delivers a write request and the LBA and length of **LUN 1** are 64 MB and 5 MB, respectively. Then the data is first divided into two slices **S2** (64 MB to 68 MB) and **S3** (68 MB to 69 MB). According to the hash algorithm, **S2** and **S3** are forwarded to **Storage node 1** and **Storage node 2** for processing, respectively. After the processing is complete, a write success message is returned to the upper-layer application.

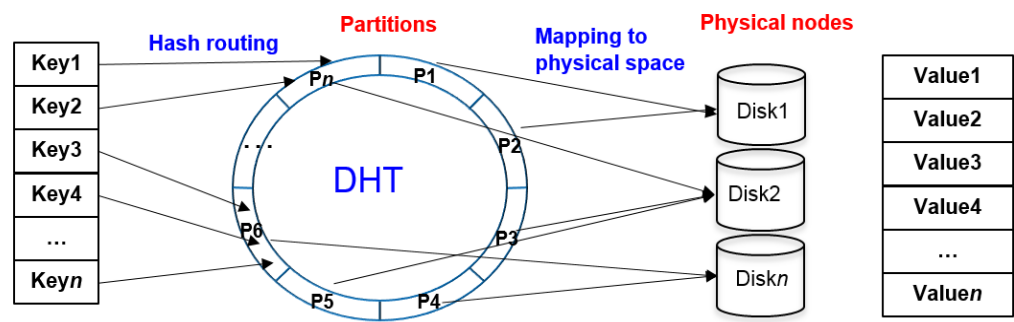
3.2.2 Back-End Load Balancing

After a request is divided through front-end loading balancing, it is processed by each home node at the service layer. For a read request, the node for data persistence needs to be located rapidly. The STORE DHT algorithm is used for fast routing. For a write request, data is finally written to the persistence layer using Plog semantics. When selecting a Plog, you need to consider back-end node and disk capacity balance. Dynamic intelligent partitioning and static disk selection algorithms are used.

3.2.2.1 STORE DHT

A key in STORE DHT consists of a Plog ID and an offset. After hash-based calculation, the system determines the partition where data is stored and maps the partition to the specific location of the disk. Partitions can be classified into cabinet-, node-, and disk-level partitions. By default, partitions are created across nodes. The system evenly allocates capacity to each partition. OceanStor Pacific series uses DHT for data addressing. The following figure shows the implementation:

Figure 3-12 STORE DHT routing



The DHT ring in OceanStor Pacific series contains 2^{32} logical space units, and is evenly divided into N partitions. The N partitions are evenly allocated on all disks in the system. For example, if N is 3600 and the system has 36 disks, each disk is allocated 100 partitions. The system configures the partition-disk mappings during system initialization and will adjust the mappings accordingly after the number of disks in the system changes. The partition-disk mapping table occupies only a small space, and nodes of OceanStor Pacific series store the mapping table in the memory for fast routing.

The following is an example:

Suppose that an application requests to access 4 KB data in a Plog ID and an offset. The system first constructs a key using a Plog ID and an offset, calculates the hash value of the key, performs the modulo operation for N , and obtains the partition ID. Then, the system locates the disk that stores the data according to the partition-disk mapping table in the memory.

NOTE

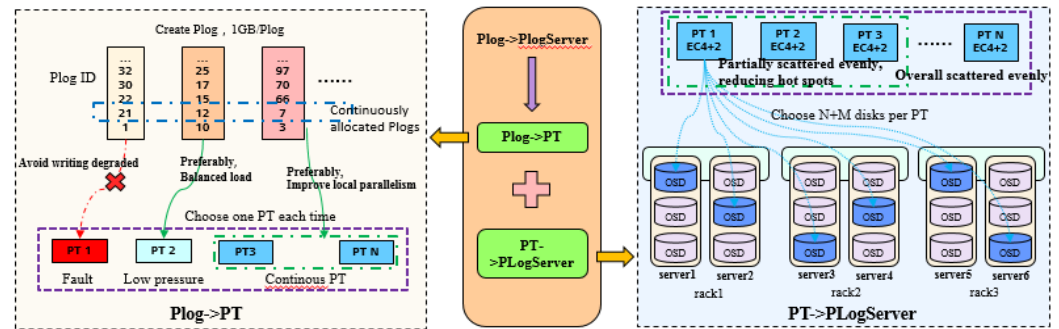
The structured service supports cabinet-level redundancy.

3.2.2.2 Dynamic Intelligent Partitioning and Static Disk Selection Algorithms

During data persistence, OceanStor Pacific series uses two algorithms to optimize the performance and reliability of scale-out storage:

- Create a Plog and select the dynamic intelligent partitioning algorithm of partitions for the Plog.
- Partitions select the local static disk selection algorithm of OSDs.

Figure 3-13 DHT dynamic intelligent partitioning and static disk selection algorithms



The dynamic intelligent partitioning algorithm introduces a self-adaptive negative feedback mechanism to achieve superb reliability and performance. Its major improvements and objectives are as follows:

- Write reliability is not degraded. If the partition corresponding to a Plog falls into a faulty disk, the Plog is discarded and a new Plog is selected to write data.
- Loads are balanced and hotspots are eliminated. In random access scenarios, polling or distributed hash algorithms cannot fully ensure balanced data layouts and disk access performance. In addition, disk fault recovery, slow disk hotspots, and QoS traffic control affect system performance. OceanStor Pacific series periodically collects statistics on the available capacity and I/O workloads of disks, nodes, and partitions to intelligently identify hotspot and slow disks.

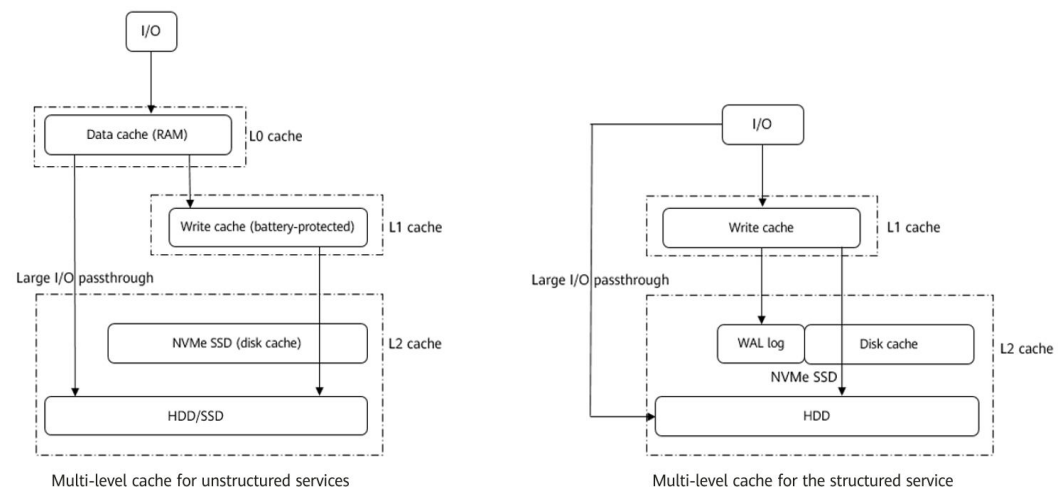
The local static disk selection algorithm aims to optimize the local balancing of the mapping between a partition and an OSD. Optimization is achieved in the following scenarios:

- Data balancing performance is optimized without affecting reliability when nodes are added for capacity expansion, reducing invalid data migration. In the algorithm, cyclic selection of partitions by partition group rather than conventional disk-based balancing is used. Intra-group balancing significantly reduces invalid data migration without degrading reliability.
- In multi-copy scenarios, the balancing of primary partitions is optimized. Common algorithms only focus on the balancing of initial primary copies. Once a node or a disk becomes faulty, primary copies are selected again for many data blocks. Whether the new primary copies are evenly distributed affects the current service performance and the performance of subsequent incremental data synchronization. The existing solution is static fixed selection of primary copies, that is, selecting secondary copy 1 first and then secondary copy 2 after a primary copy becomes faulty. OceanStor Pacific series uses dynamic selection of primary copies. Specifically, the system selects primary copies based on the busy degree of the disks where secondary copies reside. This prevents some disks from becoming hotspots after a fault occurs.
- In EC scenarios, the number of supported concurrent data rebuilding tasks is improved and performance is augmented. The correlation between partition groups and global disks is also used for small-scale calculation. The smaller the number of data blocks read from a single normal disk, the more the disks involved in load balancing.

3.3 Multi-Level Cache Acceleration

OceanStor Pacific series products leverage the multi-level cache technology to aggregate and accelerate I/Os layer by layer, improving the overall I/O performance of the system. In scenarios where both SSDs and HDDs are used, a multi-layer cache mechanism of memory, SSDs, and HDDs is constructed to reduce the I/Os written to HDDs and improve the overall performance. In all-flash configuration scenarios, a multi-layer cache mechanism of memory and SSDs is constructed to achieve ultimate performance.

Figure 3-14 Multi-level cache



In the all-flash configuration, unstructured services do not have the L2 cache, and the structured service does not have the disk cache in the L2 cache.

- **Multi-level cache for unstructured services:**

The roles of nodes of unstructured services in OceanStor Pacific series are classified into access nodes, file home nodes, and EC data persistence node. Cache acceleration is implemented for each of the three roles. As shown in the preceding figure, **L0** is located on an access node of the system, **L1** is located on a file home node, and **L2** is located on a data persistence node.

If a standard protocol is used for access, a storage node that receives requests is an access node. If the DPC protocol is used for access, a DPC node is an access node.

The L0 cache is not persistent. Asynchronous write I/Os of applications on compute nodes are directly written to the L0 cache and a success message is then returned. After an application delivers a **flush** command or the L0 cache is intelligently aggregated, the I/Os are written to the back-end storage. For details about the process, see [3.4.1 Key I/O Processes of Unstructured Data Storage Services](#).

The L1 cache is built based on the battery-protected memory and is mirrored between nodes to ensure that cache data is not lost upon a node fault or node power-off.

- **Multi-level cache for the structured service:**

The structured service in OceanStor Pacific series does not support the battery-protected memory. To implement cache acceleration, SSDs are divided into two parts in terms of functions:

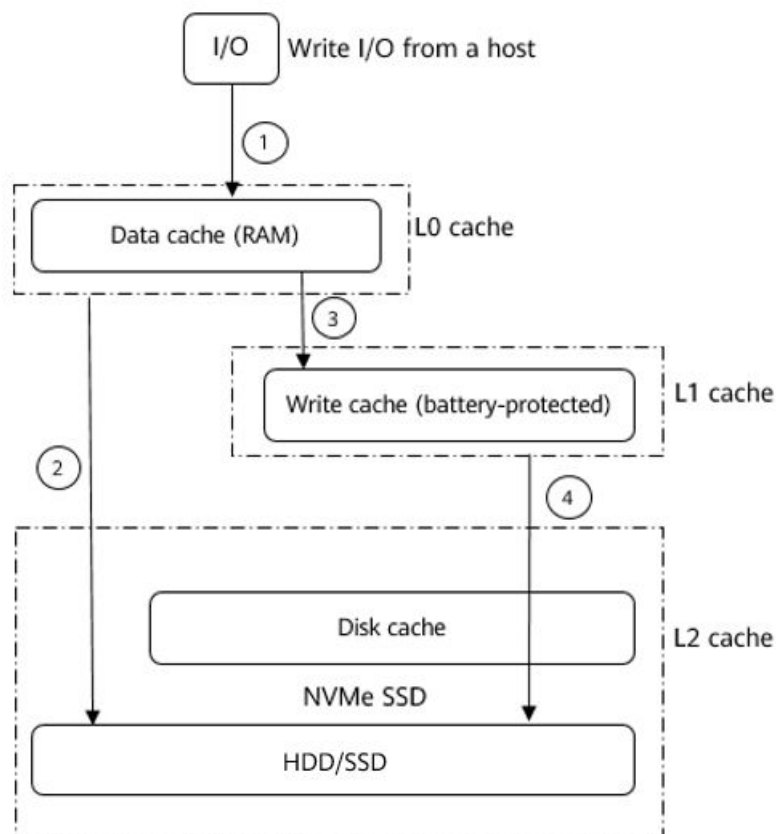
The first part is to store the L1 cache persistently, and is called WAL cache. It caches write I/Os from hosts to ensure temporary persistency of data and prevent data loss upon power failures. Small cross-node EC schemes are usually used to ensure data reliability. Small I/Os are intelligently aggregated into large I/Os in the L1 cache and written to back-end HDDs using a large EC scheme.

The second part is used as the L2 cache of HDDs, and is called disk cache. It caches data before it is written into HDDs. In this way, I/Os are not directly written to HDDs, reducing the latency. In the all-flash configuration, the disk cache will not be built.

3.3.1 Write Cache Acceleration

- **Write I/Os of unstructured services:**

Figure 3-15 Write cache of unstructured services



- Asynchronous write I/Os sent by the host are written to the data cache (RAM) of the access node and then a write success message is returned. Synchronous write I/Os will be written to the data cache in passthrough mode. The system determines whether to perform step **b** or **c** based on

the I/O size. After the data written to the data cache (RAM) reaches a certain threshold, the data is written onto disks. The system determines whether to perform step **b** or **c** based on the aggregated I/O size.

 NOTE

For data written to the data cache in asynchronous mode, in addition to the preceding flushing mechanism, related data is written onto disks when the host delivers the **flush** command based on semantic standards to ensure that data is not lost after a flush success message is returned to the user.

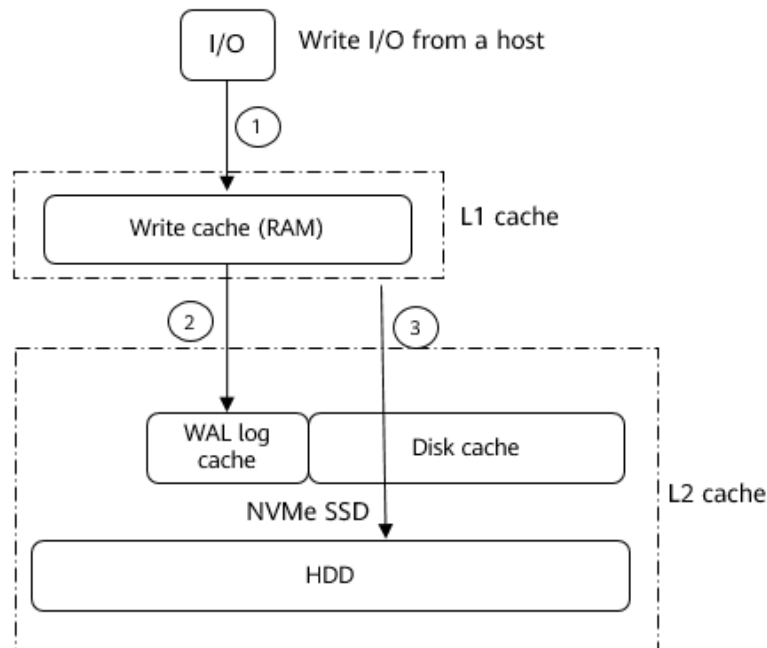
- b. If the I/Os are aggregated into large I/Os, they are calculated by EC directly. Then, the EC fragments are written to other nodes for persistency. The metadata is forwarded to the home node, written to the L1 cache, and then a write success message is returned.
- c. If I/Os are not aggregated into large I/Os, data is forwarded to the home node, written to the L1 cache, and mirrored to other nodes, ensuring that data is not lost when a node fault occurs. Then, a write success message is returned.
- d. The home node aggregates the small I/Os written to the L1 cache into large I/Os in the background until a full stripe is formed. The large I/Os are then calculated by EC, and written to other nodes for persistency. In HDD hybrid configuration scenarios, small I/Os are first divided by EC, written to persistence nodes, written to the L2 cache, and then aggregated into large I/Os. The large I/Os are then written onto HDDs. This improves the throughput of HDDs.

 NOTE

EC fragments in the L1 cache form a full stripe as much as possible. For example, if the EC scheme is 20+2 and the fragment unit is 32 KB, data in the L1 cache that has been aggregated into twenty 32 KB blocks can form a full stripe to reduce write penalty. However, directly writing 32 KB blocks to HDDs cannot maximize the throughput of the HDDs. Blocks smaller than 256 KB are aggregated in the L2 cache and then written onto HDDs when the block size exceeds 256 KB. If data in the L1 cache can be aggregated into a specification of 32 KB x 20 x 8, that is, 256 KB data will be written onto an HDD. Then, such data does not need to be aggregated in the L2 cache.

- **Write I/Os of the structured service:**

Figure 3-16 Write cache of the structured service



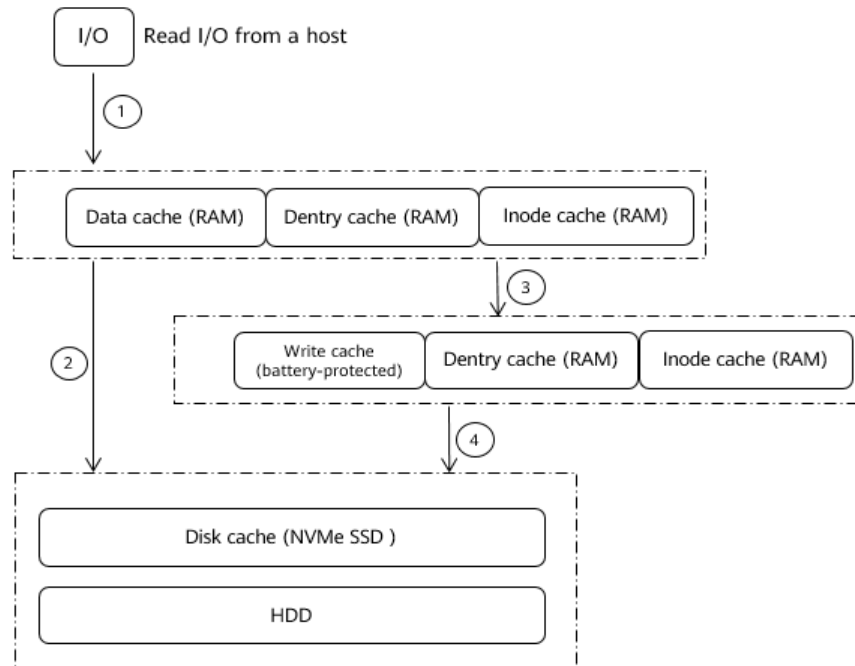
The structured service mainly uses SSDs as cache disks for acceleration, as shown in the preceding figure.

- A write I/O sent by the VBS is forwarded to the home node and written to the L1 cache. If the I/O is large, it is written in the process described in step c and then a write success message is returned.
- Data written to the L1 cache is recorded in the WAL log cache across nodes in log mode (using the fixed 2+2 small EC scheme). After the data is successfully written, a write success message is returned.
- Data in the L1 cache is sorted by I/O and written into a stripe. After the stripe is full, such data is written to another persistence node in copy or EC mode. In HDD hybrid configuration scenarios, large I/Os are directly written onto HDDs. Small I/Os are first written to the disk cache and are aggregated into large I/Os. The large I/Os are then written onto HDDs. This improves the throughput of HDDs.

3.3.2 Read Cache Acceleration

- Read I/Os of unstructured services:

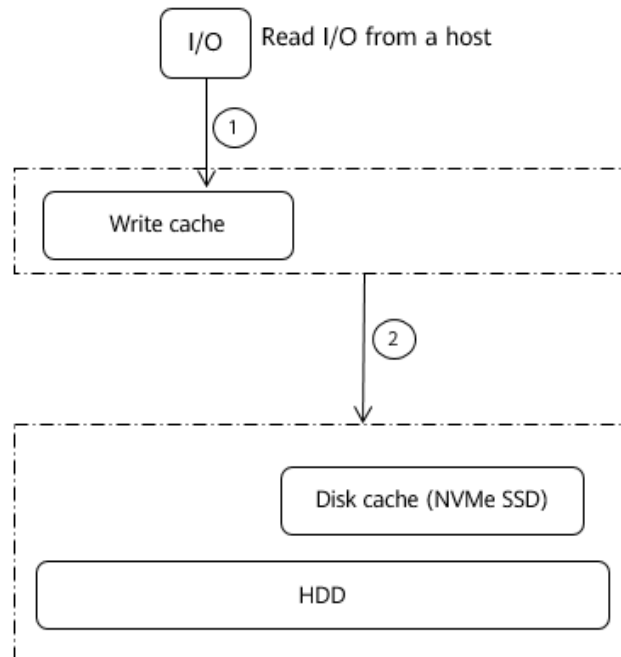
Figure 3-17 Read cache of unstructured services



- A read I/O sent from a host preferentially searches for the target data in the cache on the access node. If data is hit, the read data is returned to the host. To improve metadata performance, dentry and inode caches are rebuilt on the access node. If data is missed, the system determines whether to perform step **b** or **c** based on the read I/O size. If the metadata is missed, step **c** is performed.
- If the read I/O size is large, the data is directly read from the persistence node.
- If the read I/O size is small, it is forwarded to the home node for processing. The home node preferentially searches the write cache memory (battery-protected memory) for the data (because the data may not be written onto disks). If the data is hit, it is returned.
- If the data is missed in the write cache on the home node, the read I/O is forwarded to the persistence node. The persistence node searches the data first from the disk cache (NVMe SSDs) and then from HDDs.

- **Read I/Os of the structured service:**

Figure 3-18 Read cache of structured services



The read cache of the structured service in OceanStor Pacific series consists of two layers. The first layer is the memory cache. The second layer is the SSD cache, which functions based on the hotspot read mechanism. The system collects statistics on each piece of read data and the hotspot access factors. When the threshold is reached, the system automatically caches data to SSDs and removes the data that has not been accessed for a long time from SSDs.

When a storage node receives a read I/O operation sent by the VBS, the read I/O is processed as follows:

- a. The node checks whether the required I/O data is in the write cache. If yes, the data is returned. Otherwise, step **b** is performed.
- b. The read request is forwarded to an OSD node, then the node checks whether the required I/O data exists in the SSD cache. If yes, the data is returned. Otherwise, the OSD node searches the disks for the required I/O data and returns the data. In addition, the hotspot access factors of the I/O data are increased. If the hotspot access factors reach the threshold, the I/O data is cached in the SSD cache.

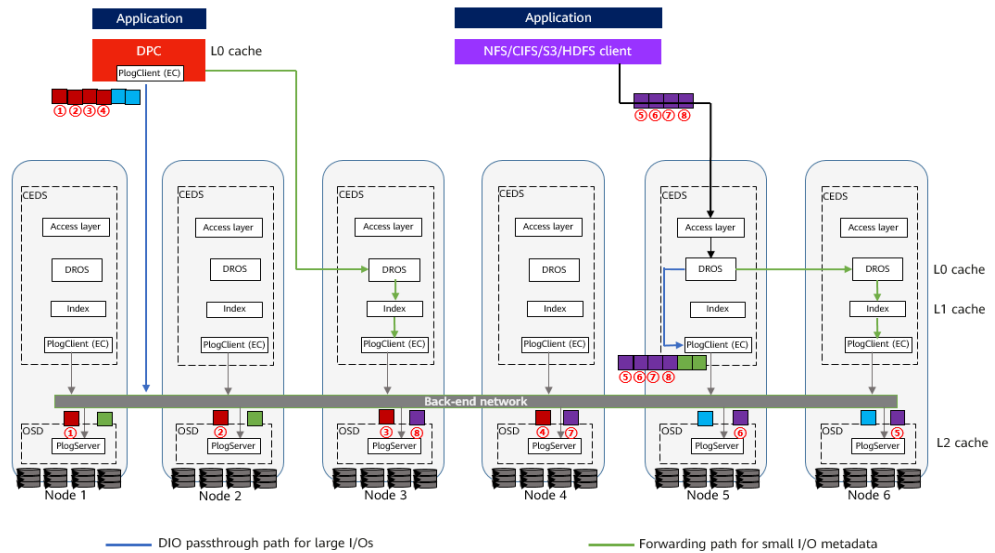
3.4 Key I/O Processes

3.4.1 Key I/O Processes of Unstructured Data Storage Services

DROS is the management platform of unstructured data in OceanStor Pacific series. It provides unstructured services and normalizes basic I/Os. To reduce network overheads, large read and write I/Os are processed in DIO passthrough mode. In this way, large data blocks do not need to be forwarded to the home

node and then written to the persistence layer. Instead, data is directly accessed from the access node to the persistence layer.

Figure 3-19 I/O process of unstructured services



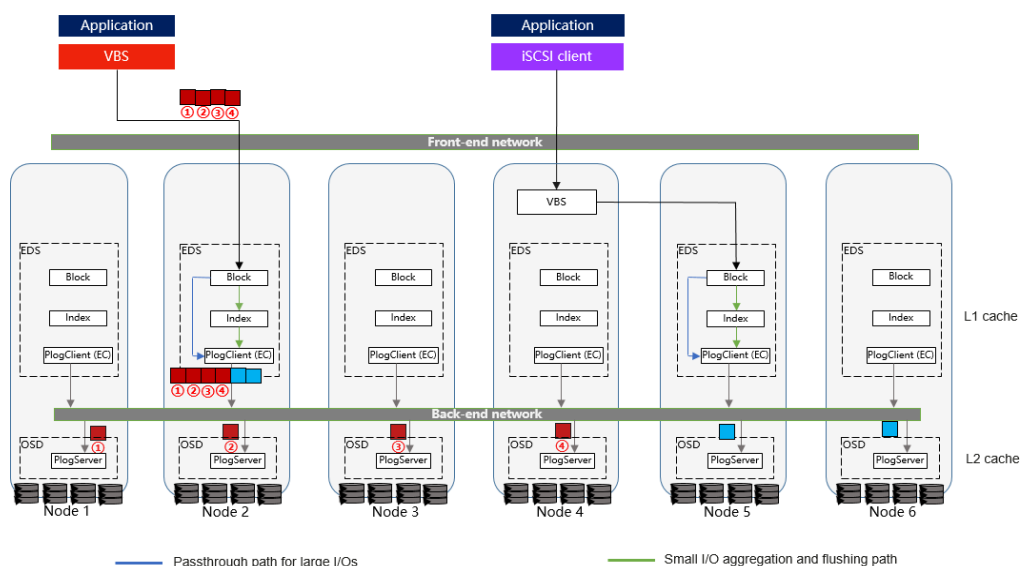
As shown in the preceding figure, when a DPC node functions as an access node, asynchronous I/O data blocks 1, 2, 3, and 4 are written to the L0 cache on the DPC node. When data is flushed onto disks from the L0 cache, if the data blocks are aggregated into a large I/O in the L0 cache, a full stripe is calculated by EC and directly written to the OSDs in passthrough mode. Then, the metadata is forwarded to the home node. For small I/Os, they are forwarded to the L1 cache on the home node for processing directly. Small I/Os in the L1 cache are aggregated into large I/Os, calculated by EC, and then written to the OSDs. When a standard protocol is used for access, storage nodes function as access nodes. Asynchronous write I/O data blocks 5, 6, 7, and 8 are written into the L0 cache. The data blocks are aggregated into a large I/O in the L0 cache, calculated by EC, and then written to the OSDs. Then, the metadata is forwarded to the home node and written to the L1 cache. For small I/Os, they are forwarded to the CEDS of the home node and written to the L1 cache, and wait to be aggregated and written onto disks.

NOTE

Metadata operations, such as creation, deletion, and attribute setting, are small I/Os and are directly forwarded to the home node for processing.

3.4.2 Key I/O Processes of the Structured Data Storage Service

Figure 3-20 I/O process of the structured service



An upper-layer application sends a write I/O request to VBS of the storage system. VBS forwards the request to a specific storage node calculated using the DHT hash algorithm at layer 1. Then EDS (block + index) on the node will process the request.

After receiving the request, EDS writes the data to the L1 cache using a small EC scheme. In addition, EDS stores the data in the memory of the storage node where the EDS resides. After that, EDS returns a write success message to VBS, and VBS forwards the message to the upper-layer application. After the data in the L1 cache memory is aggregated into a larger block and is aggregated into a full stripe, it is calculated by EC, and then written to the OSD through the Plog interface. Large I/Os do not need to be aggregated in the L1 cache. They are directly calculated by the EC ratio configured by the user and then written to the OSD.

NOTE

The VBS of the structured service can be deployed on storage nodes or compute nodes.

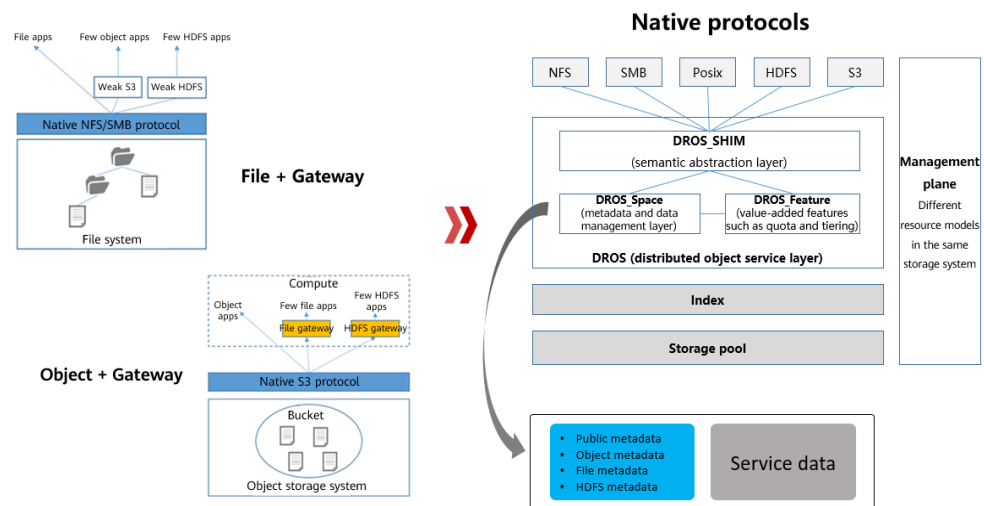
3.5 Convergence and Interworking of Unstructured Data Storage Services

Internet of Everything (IoE), cloud, and intelligence drive the rapid growth of data, among which unstructured data occupies a large proportion. However, due to the differences of services such as file, object, and HDFS, multiple services are constructed in silo mode. Users need to use multiple storage systems at the same time so that full-lifecycle management of data can be implemented. Take autonomous driving as an example. Different types of data need to be accessed in different processing phases. The end-to-end data processing process requires

multiple protocols or multiple storage systems. In such solutions, users have the following pain points:

- **Space waste:** The same data needs to be stored in multiple storage systems, wasting storage space and increasing storage costs.
- **Low efficiency:** The same data needs to be replicated across multiple storage systems. The performance of a common object storage system is poor. As the data volume increases, the replication may take several hours, several days, or even longer, greatly affecting data processing efficiency. In the field of autonomous driving mentioned above, the time consumed by replication will directly affect the R&D efficiency of automobiles.
- **Difficult O&M:** Different device management and system operations of multiple storage systems greatly increase the O&M difficulty of management personnel.

Figure 3-21 Convergence and interworking of unstructured services in OceanStor Pacific series



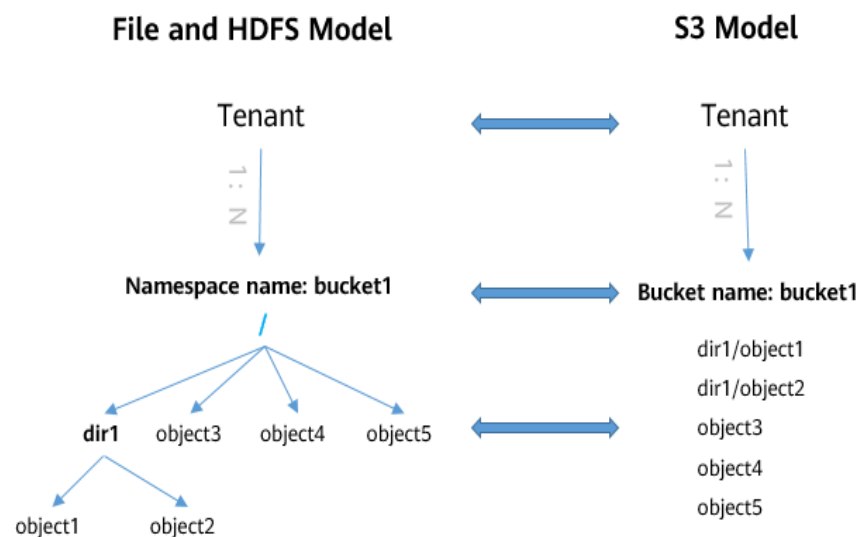
In the industry, there are two types of architectures that support convergence and interworking of unstructured data: interworking architecture based on a file storage system and interworking architecture based on an object storage system. Both the two architectures support data interworking based on basic I/O semantics. The metadata and data management system can be used for only one type of ecosystem applications and has only one type of native semantics. The other type is not supported or the protocol is weak, which cannot support upper-layer ecosystem applications. To solve these problems, OceanStor Pacific series launches a technical solution that converges and interconnects three unstructured services: file, object, and HDFS. The solution has the following key capabilities:

- **Unified management plane:** unified resource models, such as tenants/users, networks, and storage pools
- **Shared storage pools:** Unstructured services are provided by the same storage pool. Multiple services share the same data.
- **I/O semantic interworking:** full native semantics, lossless protocols, and no external plug-ins

- **Value-added feature interworking:** Unstructured services share and interconnect advanced features, such as QoS, tiering, and quota, without semantic loss. The scalability, performance, and serviceability are unified.

Unstructured services of OceanStor Pacific series share storage pools. This is the prerequisite for semantic interworking among multiple services. One piece of data can be accessed through multiple protocols. In order to implement semantic interworking, the conceptual models of multiple unstructured services are unified. Unstructured services of OceanStor Pacific series provide namespaces in a unified manner. In the file service, a namespace equals to a file system. In the object service, a namespace equals to a bucket. User data in each namespace is isolated from each other. System resource contention is controlled by the SmartQoS mechanism. One namespace can support HDFS, file, and object services at the same time.

Figure 3-22 Mappings between namespaces and buckets



Based on the DROS base and to implement unstructured service data interworking, namespaces are mapped as buckets of the object service. A directory structure of the file service is mapped to a URL of the object service, as shown in the preceding figure. The namespace is mapped to **bucket1**, files **object1** and **object2** in directory **/dir1** are mapped to **dir1/object1** and **dir1/object2** in bucket **bucket1**, and files **object3**, **object4**, and **object5** in the root directory are mapped to objects **object3**, **object4**, and **object5** in bucket **bucket1**.

Alternatively, objects created through the S3 protocol are mapped to the corresponding directory structures. If an empty namespace is mapped to bucket **bucket1**, DROS will create directory **dir1** and file **object3** in directory **dir1** for object **dir1/object3** in bucket **bucket1** created through S3. In addition, for object **object3** in bucket **bucket1** created through S3, DROS will create file **object3** in the root directory of the corresponding namespace.

In this way, the data written in the file service can be accessed by the object service, and the data written in the object service can also be accessed by the file service.

NOTE

In the preceding example, there are two levels of file directories. The mappings between multi-level directories are the same. For example, file **/dir1/dir2/object** will be mapped to object **/dir1/dir2/object** in bucket **bucket1**.

The unified basic model can only solve the basic problem of multi-service interworking and access. The service semantics of file, HDFS, and object services are different, such as locks and permissions. Except the basic mapping model, there are three key points for I/O semantic interworking:

- **Lossless semantic interworking**

S3 interfaces such as **ListObjects**, multipart, and versioning

Concat interface of the HDFS protocol

- **Permission interworking**

When a file is accessed by multiple protocols at the same time, how to implement permission interworking to ensure data security?

- **Lock interworking**

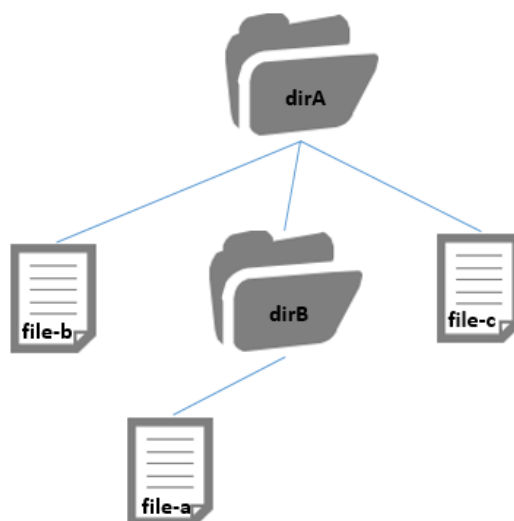
When a file is accessed by multiple protocols, how can the locks of different protocols interwork with each other and how can data consistency be ensured upon concurrent accesses?

3.5.1 Lossless Semantic Interworking

3.5.1.1 ListObjects Interface of the S3 Protocol

The **ListObjects** interface of the S3 protocol is similar to the **readdir** interface of the NAS and HDFS protocols. However, the **ListObjects** interface of the S3 protocol provides more powerful functions, such as listing results in the alphabetical order and searching and paging results by prefix or mark. In addition, this interface does not have directory semantics. Instead, it lists the full path of each file as the results.

Figure 3-23 Traditional directory metadata structure



As shown in the preceding figure, directory **dirA**, sub-directory **dirB**, and files **file-a**, **file-b**, and **file-c** are created using the NFS protocol. If the NFS protocol is used to invoke the **readdir** interface of directory **dirA**, the return values are as follows:

file-b

dirB

file-c

The returned results of the **readdir** interface invoked by protocols such as NFS are not in order, and only the sub-directories and files in the current directory are returned.

If the S3 protocol is used to invoke the **ListObjects** interface and **prefix** is set to **dirA**, the return values are as follows:

dirA/dirB/file-a

dirA/file-b

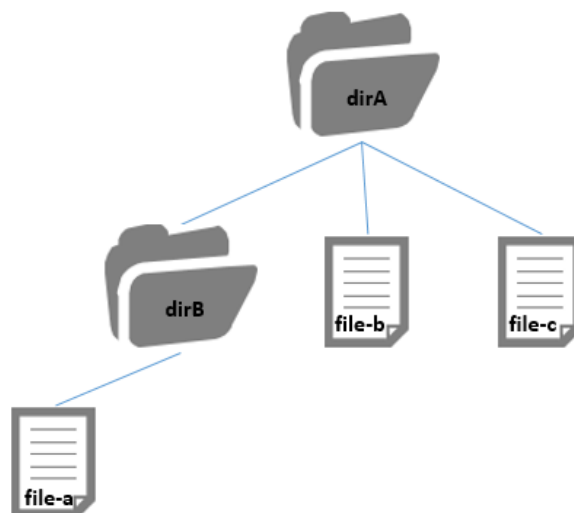
dirA/file-c

The results show that the full paths are returned when the S3 protocol is used. In addition, the results are sorted in the alphabetical order.

For vendors that provide traditional interworking, directory metadata generally uses the globally disordered structure and does not support the **ListObjects** interface of the standard S3 protocol.

Based on these requirements, the directory metadata of OceanStor Pacific series can be stored in order. The following figure shows the metadata structure in an OceanStor Pacific series storage system:

Figure 3-24 Directory metadata structure of OceanStor Pacific series



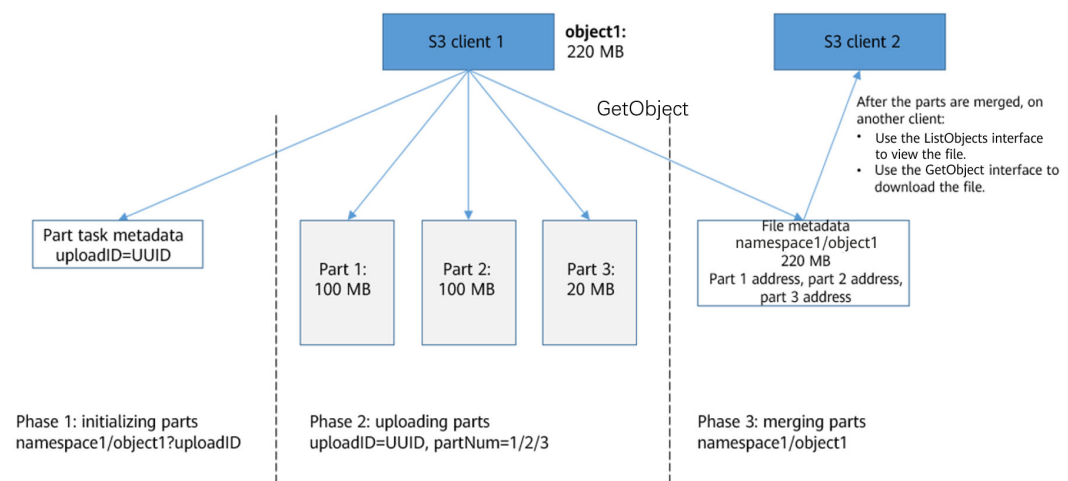
Files written using any protocol, such as NFS and SMB, support this metadata layout. This ensures that OceanStor Pacific series can provide the capabilities of the native **ListObjects** interface in multi-protocol interworking scenarios and is compatible with various function parameters of the interface.

3.5.1.2 Multipart Interface of the S3 Protocol

The multipart upload interface is an extremely important interface for the S3 protocol. Almost all S3 applications use the multipart upload interface to upload medium- and large-sized files of several MB. Therefore, if multipart interface interworking is not supported, S3 protocol interworking is not supported.

However, this interface is not supported by other protocols and no similar interface can be found. The principle of the S3 multipart interface is as follows:

Figure 3-25 Principle of the S3 multipart interface



Initializing parts: Generates a UUID of a multipart task.

Uploading parts: Uploads data by part number. In this process, the parts can be uploaded using multiple client threads or multiple clients concurrently, improving upload performance. For example, a client has a 220 MB file and the file is uploaded using three threads. Thread 1 uploads 0 to 100 MB, thread 2 uploads 100 to 200 MB, and thread 3 uploads 200 to 220 MB. The three threads do not affect each other. If a part fails to be uploaded, the client only needs to retry the failed part.

Merging parts: After all parts are uploaded, the client merges the parts. The system generates file metadata of **object1**, which records the parts that comprise the file. In this step, only metadata operations are performed, and data is not merged. Therefore, the efficiency is high.

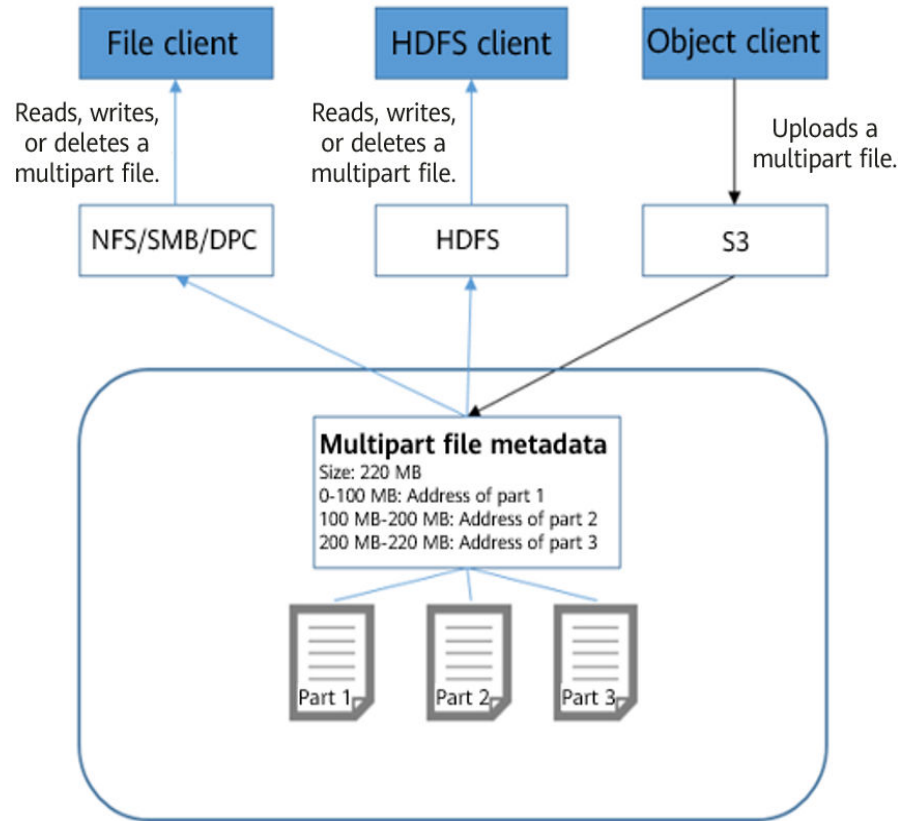
The file is visible to external users and can be downloaded or deleted by other clients only after parts are merged in phase 3.

According to the preceding description, the multipart interface of the S3 protocol not only supports high-concurrency and high-performance upload on clients, but also provides functions similar to resumable upload. Different parts do not affect each other. If an upload fails, only the part that fails to be uploaded is retried.

Generally, interworking architectures in the industry do not support cross-protocol access of multipart files, or only supports multipart file read over other protocols. Therefore, it cannot be considered that the interworking architectures support the S3 protocol.

The multipart interface of OceanStor Pacific series allows multipart files to be read and written using all other protocols. The following figure shows the implementation:

Figure 3-26 Multipart processing



As shown in the preceding figure, for a multipart file, DROS records the special metadata structure as well as the part file address corresponding to each offset of the file. The implementation principles of the main operations of other protocols are as follows:

- Offset read operation: An offset is converted into an offset read in the corresponding part.
- Modification operation: An offset is converted into an offset write in the corresponding part.
- Truncate operation: The part data corresponding to a truncate offset is recycled.

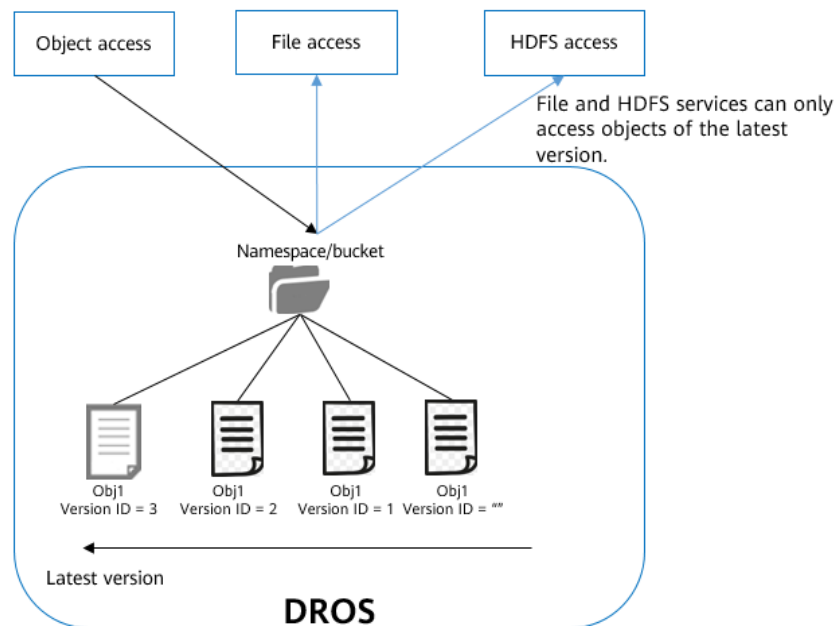
3.5.1.3 Versioning Interface of the S3 Interface

S3 versioning is an important interface for objects. With versioning, objects of multiple versions can be retained in the same bucket, so that users can easily search for and restore objects of different versions. In this way, data can be quickly restored in the case of misoperations or application faults.

The versioning interface of OceanStor Pacific series can be enabled or disabled by bucket and is disabled by default. The interface can be enabled or disabled using an API or on the management console. If this interface is disabled, the version ID

of each object to be uploaded is empty, and the object with the same name will be overwritten during the upload. With this interface enabled, each uploaded object has a non-empty version ID. If an object to be uploaded is with the same name as an uploaded object, a new version is generated and the existing object will not be overwritten. In HDFS and file access, only file names and data of the latest versions are displayed.

Figure 3-27 Versioning processing principle



Once versioning is enabled, it takes effect permanently. However, you can switch it between the enabled and suspended status.

3.5.2 Permission Interworking

The security of cross-protocol access among NFS/SMB/DPC/FTP/FTPS, HDFS, and S3 protocols must be ensured. Namely, permission information can be exchanged between the protocols and the permissions of users of each protocol can be controlled precisely. For example, the permissions configured for SMB users also apply to NFS users. However, users of different protocols belong to different domains (for example, SMB users belong to the AD domain while NFS users belong to the LDAP/NIS domain), and have different user identification ways (an SMB user is defined by SID while an NFS user by UID/GID). As a result, the configured permissions are hard to be exchanged (permissions are managed by users/user groups).

The following focuses on the interworking among the SMB, NFS, and S3 protocols. The user management and permission management of HDFS/DPC/FTP/FTPS are basically the same as those of NFS.

3.5.2.1 NFS and SMB

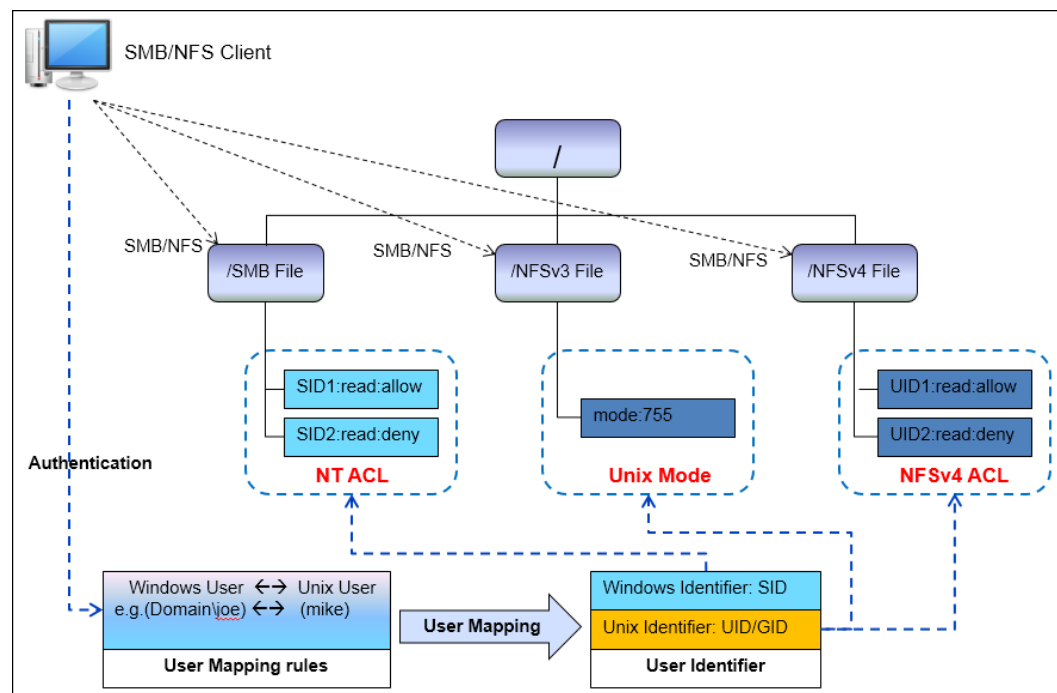
To achieve permission interworking, users of different protocols need to be associated. OceanStor Pacific series employs user mappings (mapping between a

Windows user and a UNIX user) to achieve this goal. In the user mapping solution, an ID that applies to both SMB and NFS protocols includes the following parts:

- User SID and user group SID, used for SMB share permission authentication and NT ACL authentication
- User UID and user group GID, used for UGO authentication

After an SMB or NFS user obtains such an ID based on user mapping, one part of the ID (such as user SID) is selected based on the specified ACL type in the ACL authentication phase, as shown in the following figure:

Figure 3-28 User mapping for SMB-NFS cross-protocol access



In an SMB-NFS cross-protocol access service, minimize the impact on file I/O performance when user mapping is performed. Try to perform user mapping on non-I/O data flows. The differences between different protocols are as follows:

- In the SMB protocol, for an SMB session, only one user authentication action is performed during session establishment to confirm user permissions, which are used for subsequent file access authentication of the session. Therefore, in SMB services, user mapping is performed during user authentication.
- In the NFS protocol, the client sends user information to the storage system for each file operation request. Different from the SMB protocol, user authentication is not valid for other operations. Therefore, user mapping is performed during ACL authentication. User mapping is required only when the ACL of a file is NT ACL, which reduces unnecessary mapping operations.

Permission-controlled behaviors:

You can set different security styles for a file system or a dtree to meet permission control and management requirements in different application scenarios. For example, if a file system is mainly used for NFS service access and the SMB service is used only for simple read services, you can set the security style of the file system to UNIX and use UNIX mode bits or POSIX

ACLs for file permission control. If a file system is mainly used for SMB service access and the NFS service is used only for simple read services, you can set the security style of the file system to NTFS and use NT ACLs for file permission control.

Currently, OceanStor Pacific series storage products support UNIX, mixed, NTFS, and native security styles. You can select a security style based on the client type and application scenario.

Style Name	Introduction
UNIX	Applies to the scenario where UNIX clients are mainly used. In this scenario, UNIX mode bits, POSIX ACLs, or NFSv4 ACLs are used for permission control. In the UNIX style, a Windows client can be used for access through the SMB protocol, but NT ACLs cannot be used for permission control (NT ACLs cannot be used for permission setting and permission check).
Mixed	Applies to permission control scenarios of CIFS clients (using the CIFS protocol) and UNIX clients (using the NFS, DPC, or HDFS protocol). UNIX mode bits, POSIX ACLs, NFSv4 ACLs, and NT ACLs can be used for permission control. In this style, CIFS permissions (NT ACLs) and UNIX permissions (UNIX mode bits, POSIX ACLs, or NFSv4 ACLs) cannot co-exist. Only the latest configured permissions on either CIFS or UNIX clients take effect.
NTFS	Applies to the scenario where CIFS clients are mainly used. In this scenario, Windows NT ACLs are used for permission control. In the NTFS style, a UNIX client can be used for access through the NFS, DPC, or HDFS protocol, but UNIX mode bits, POSIX ACLs, and NFSv4 ACLs cannot be used for permission setting.
Native	Applies to permission control scenarios of CIFS clients (using the CIFS protocol) and UNIX clients (using the NFS, DPC, or HDFS protocol). UNIX mode bits, POSIX ACLs, NFSv4 ACLs, and NT ACLs can be used for permission control. In this style, CIFS permissions (NT ACLs) and UNIX permissions (UNIX mode bits, POSIX ACLs, or NFSv4 ACLs) can co-exist, and Windows NT ACL permissions and UNIX permissions will neither affect nor synchronize with each other.

Introduction to cross-protocol access

In cross-protocol scenarios, the current protocol may access files or directories created by other protocols, and the files or directories created by other protocols have ACLs different from those of the current protocol. Cross-protocol ACL authentication determines the ACL authentication method based on the ACL type and security style of the current file. In addition, the security style restricts the access of different types of ACL permissions.

- UNIX

- Only UNIX ACLs (including POSIX ACLs and NFSv4 ACLs) and UNIX mode bits can be used for permission access and setting. NT ACLs cannot be used for permission access and setting. Even if a file has an NT ACL, the NT ACL cannot be used for authentication in the UNIX style. Instead, only UNIX mode bits can be used for authentication on the file. For example, if a user creates a file in the mixed style (the mixed style supports permission setting using NT ACLs) on a Windows client, the file has an NT ACL. Then, if the user switches the mixed style to the UNIX style and uses an NFS client to access the file, the NT ACL will become invalid and only UNIX mode bits can be used for authentication on the file.
- If the UNIX ACL of an access protocol does not match that of a share file, the UNIX ACL of the share file is used for authentication. For example, if the NFSv3 protocol is used to access a file that has an NFSv4 ACL and the NFSv3 protocol supports POSIX ACLs, the NFSv4 ACL of the share file is used for authentication.
- In the UNIX style, when a Windows client is used to access a file or directory, a Windows user must be mapped to a UNIX user for authentication using UNIX ACLs or UNIX mode bits. (The system maps a Windows user to nobody by default.)
- Mixed
 - In this style, POSIX ACLs, NFSv4 ACLs, NT ACLs, and UNIX mode bits can be used for permission access and setting. Different types of ACLs, however, do not exist at the same time. The last configured ACL takes effect.
 - In the mixed style, permissions are checked based on the ACL type of the current file or directory.
 - When a UNIX client is used to access a file created on a Windows client, user mapping is required. A UNIX user is mapped to the corresponding Windows user for NT ACL authentication.
 - When a Windows client is used to access a file created on a UNIX client, user mapping is required. A Windows user is mapped to the corresponding UNIX user for UNIX ACL authentication. For example, when you use a Windows client to mount a CIFS share to access a file or directory created by NFSv3, the file or directory may have a POSIX ACL. In this case, you need to map a Windows user to a UNIX user and then perform POSIX ACL authentication.
- NTFS
 - In this style, NT ACLs can be used for permission access and setting, UNIX mode bits can be used for permission access, and UNIX mode bits, POSIX ACLs, and NFSv4 ACLs cannot be used for permission setting. Even if a file has a UNIX ACL (POSIX ACL or NFSv4 ACL), the UNIX ACL cannot be used for authentication in the NTFS style. Instead, only UNIX mode bits can be used for authentication on the file. For example, if a user creates a file in the mixed style (the mixed style supports permission setting using UNIX ACLs) on a UNIX client,

the file has a UNIX ACL. Then, if the user switches the mixed style to the NTFS style and uses a UNIX client to access the file, the UNIX ACL will become invalid and only UNIX mode bits can be used for authentication on the file.

- If a share file has an NT ACL, the NT ACL is used for authentication. For example, if the NFSv3 protocol is used to access a file and the file has an NT ACL, a UNIX user is mapped to a Windows user and the NT ACL of the share file is used for authentication.
- Native
 - In this style, POSIX ACLs, NFSv4 ACLs, NT ACLs, and UNIX mode bits can be used for permission access and setting. Windows NT ACL permissions and UNIX permissions will neither affect nor synchronize with each other.
 - In the native style, permissions are checked based on the access protocol and the ACL type of the current file or directory.
 - When a UNIX client is used to access a file created on a Windows client, user mapping is not required. In this case, UNIX mode bits, POSIX ACLs, or NFSv4 ACLs are used for authentication.
 - When a Windows client is used to access a file created on a UNIX client and the file does not have an NT ACL, user mapping is required. A Windows user is mapped to the corresponding UNIX user for authentication using UNIX mode bits or UNIX ACLs. For example, if the security style of a file system is switched from UNIX or mixed to native and a Windows client is used to access a file that has a POSIX ACL, a Windows user needs to be mapped to a UNIX user and then authentication is performed using the POSIX ACL.

3.5.2.2 S3 and Other Protocols

Different from other protocols, the S3 protocol does not use UGO authentication or user-level ACL authentication. The S3 protocol supports the following permission control modes:

Table 3-3 S3 protocol permission control

Control Object	Permission Mechanism	Remarks
User	IAM policy	Grant specified permissions to specified users and user groups.

Control Object	Permission Mechanism	Remarks
Bucket (Corresponding to namespaces in OceanStor Pacific series)	1. Bucket ACL 2. Bucket policy	Bucket ACLs are used to control basic read, write, and read/write permissions across tenants. Bucket policies provide fine-grained permission control. Bucket policies can control objects with specific prefixes, IP address segments, or HTTPS.
Object	Object ACL	Different from other protocols, ACLs of the S3 protocol support cross-tenant authorization but does not support authorization between users in a tenant.

Based on these differences, OceanStor Pacific series can map S3 users to UNIX users for permission interworking. The implementation is as follows:

1. An administrator creates the mapping from an S3 user to a UNIX user.
2. After the mapping is successful, all files written by the S3 protocol record correct UIDs and GIDs. In this case, when a user accesses the files using another protocol, UGO authentication or POSIX ACL authentication is supported.

Figure 3-29 User mapping model

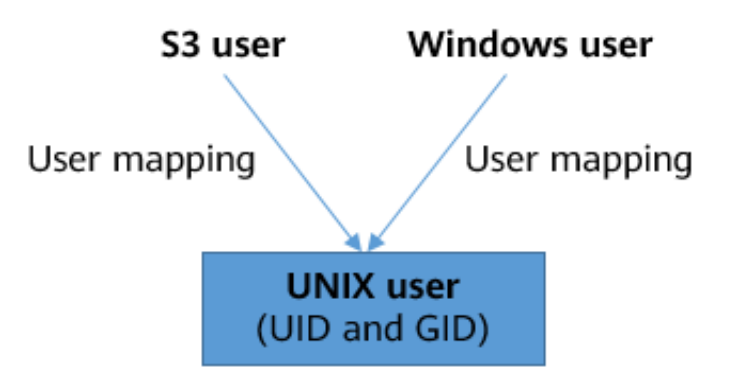
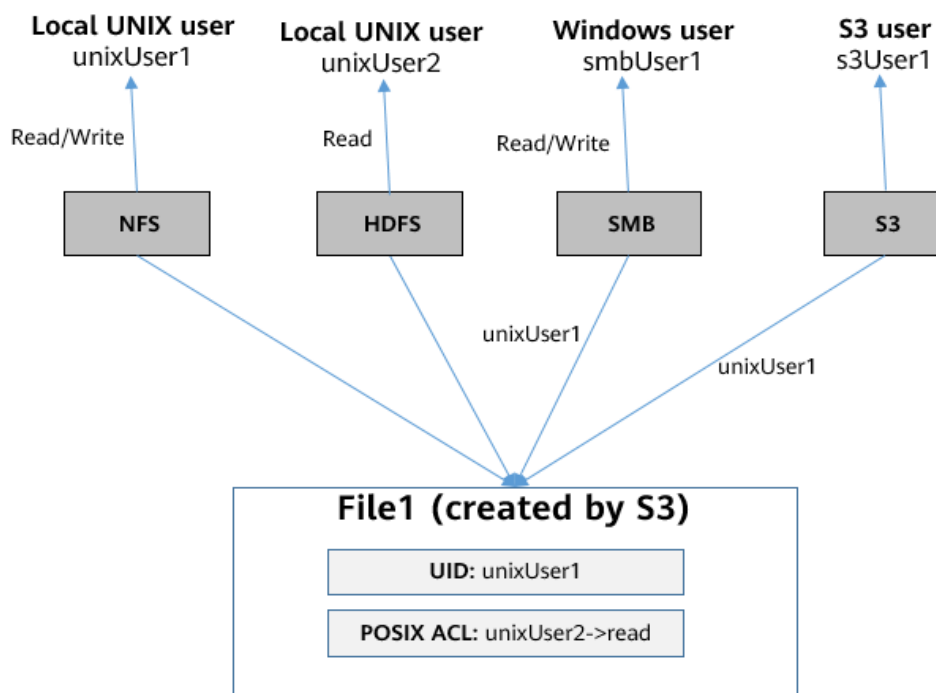


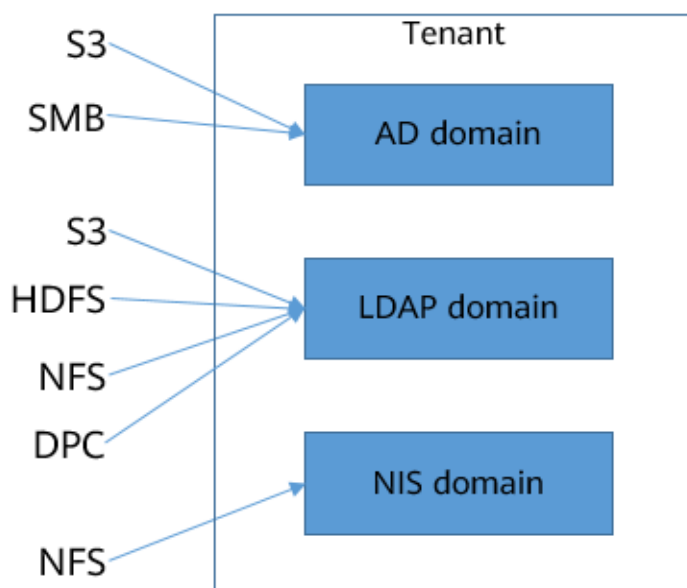
Figure 3-30 Interworking between S3 and other protocols

As shown in the preceding figure, both the S3 user and Windows user are mapped as user **unixUser1**. The UID of the file written using the S3 protocol is **unixUser1**. Then a user using the NFS or SMB protocol can correctly read and write the file. In addition, the HDFS protocol can set an ACL for the file so that user **unixUser2** can read the file. After the setting is complete, user **unixUser2** of the HDFS protocol can also read the file. In this way, permission interworking is implemented among multiple protocols.

3.5.2.3 Domain Users

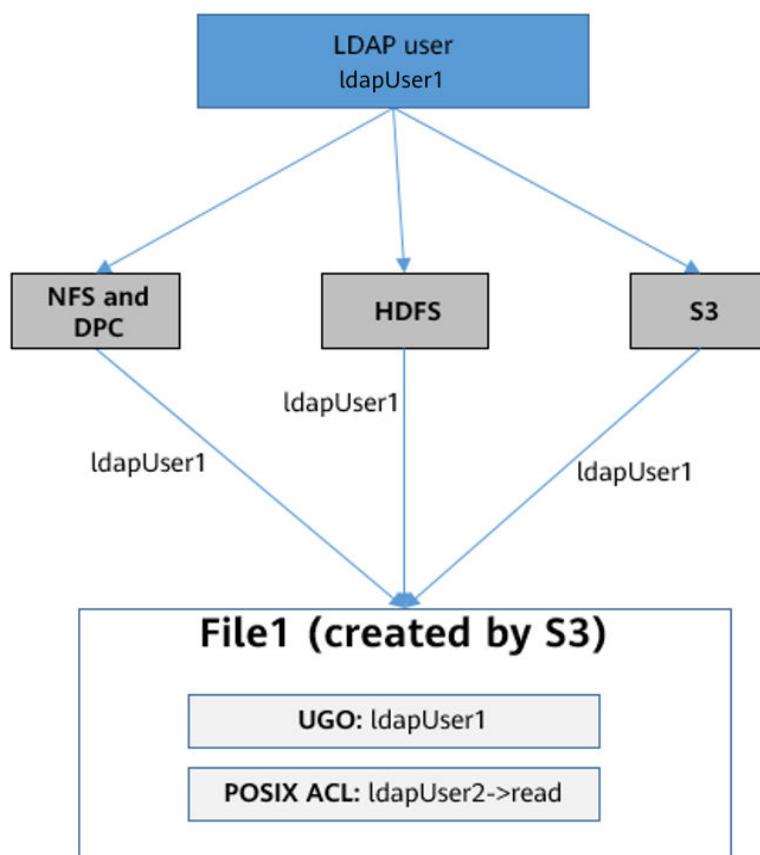
The standard NFS, SMB, HDFS, DPC, and S3 protocols in the industry support different domain authentication services. OceanStor Pacific series supports the following domain authentication capabilities:

Figure 3-31 Domain users of different protocols



As shown in the preceding figure, LDAP domain users of the same tenant support S3, HDFS, NFS, and DPC. Therefore, for LDAP domain users, OceanStor Pacific series supports permission interworking among these protocols without user mapping.

Figure 3-32 Permission interworking for LDAP domain users



As shown in the preceding figure, the S3 protocol uses LDAP domain user **ldapUser1** to create file **File1**. The UID of the file is **ldapUser1**. In this way, when the file client and HDFS client both use user **ldapUser1** to access the file, they have the corresponding permissions and user mapping is not required.

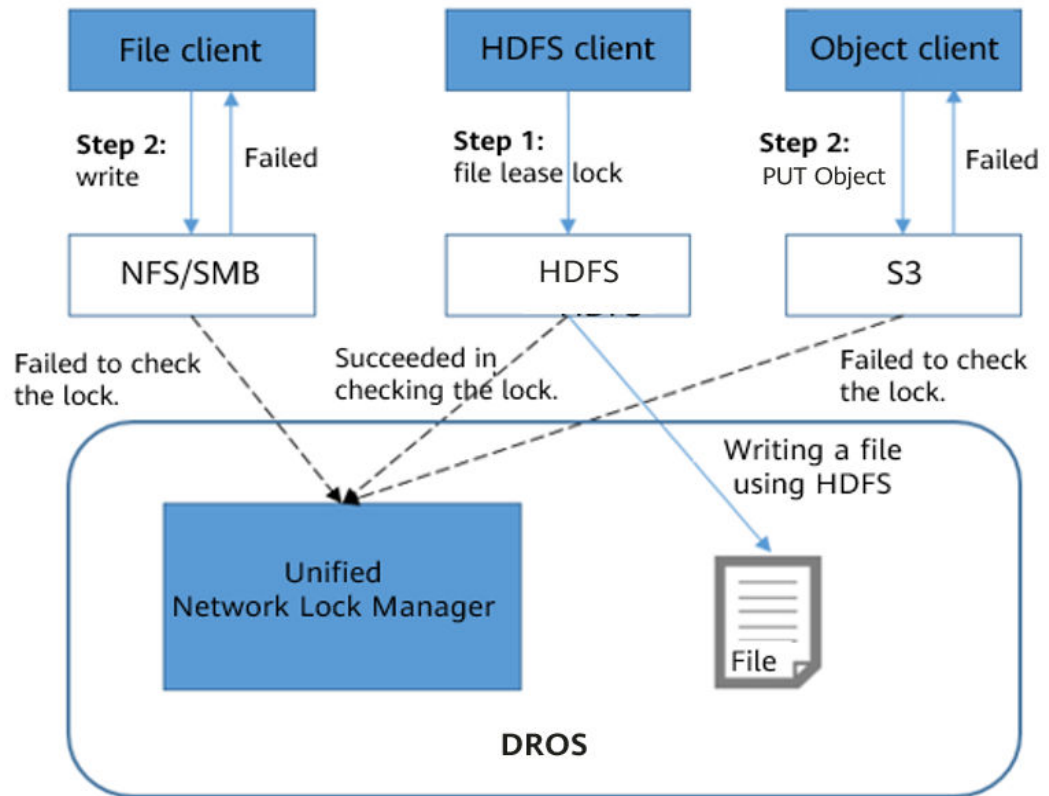
If a Windows user accesses the file using the SMB protocol, the Windows user must be mapped to the LDAP domain user.

3.5.3 Lock Interworking

OceanStor Pacific series extends the lock capabilities of the S3 and HDFS protocols. Once a mandatory lock is configured successfully for any protocol (supporting file-level and byte range locking (BRL)-level locks), other protocols will fail to be used to modify or append data. This ensures data consistency during concurrent accesses of multiple protocols.

OceanStor Pacific series provides the unified distributed lock management service NLM. Whether a lock is configured successfully is checked each time when data is concurrently accessed by multiple protocols.

Figure 3-33 Lock interworking principle

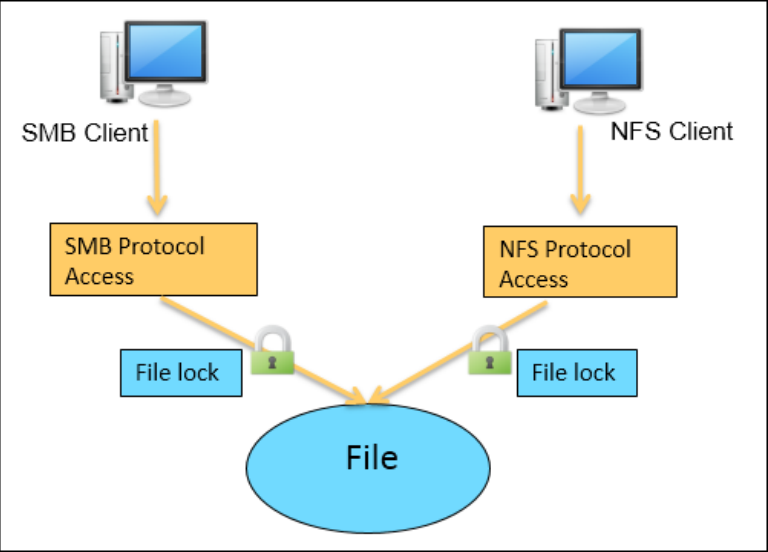


As shown in the preceding figure, the NFS/SMB, HDFS, and S3 protocols concurrently operate the same file. If the HDFS protocol first successfully applies for the lease lock, other protocols need to check whether the file is locked when modifying the same file. If the file is locked, other protocols will fail to check the lock, and the I/O will return a failure message. After the HDFS protocol succeeds in configuring a lock, the file can be modified.

SMB supports file locks but NFSv3 does not. Although both SMB and NFS support range locks, they are not mutually exclusive with file locks. Based on different application scenarios, different policies are used for mutual exclusion of file locks of different protocols:

- If the security style of a file system is UNIX, the NFS protocol is mainly used to generate data. In this mode, the NFSv3 protocol does not proactively trigger file lock detection. Lock exclusion detection is performed only when the NFSv3 protocol proactively configures a lock.
- If the security style of a file system is NTFS, the SMB protocol is mainly used to generate data. In this mode, the NFSv3 protocol proactively triggers file lock detection when opening files, as shown in the following figure.

Figure 3-34 Mutual exclusion between NFS and SMB locks



In the current product version, the lock exclusion check is performed only a service proactively configures a lock. The lock exclusion check in cross-protocol scenarios is as follows:

Table 3-4 Lock exclusion check

Lock 1	Lock 2	Check Item
NLM	NLM	Owner + Type+ Range
NLM	SMB OPEN	Type + Share access
NLM	SMB BRL	Type + Range
SMB OPEN	NLM	Type + Share access
SMB OPEN	SMB OPEN	Desire access + Share access
SMB BRL	NLM	Type + Range
SMB BRL	SMB BRL	Type + Share access

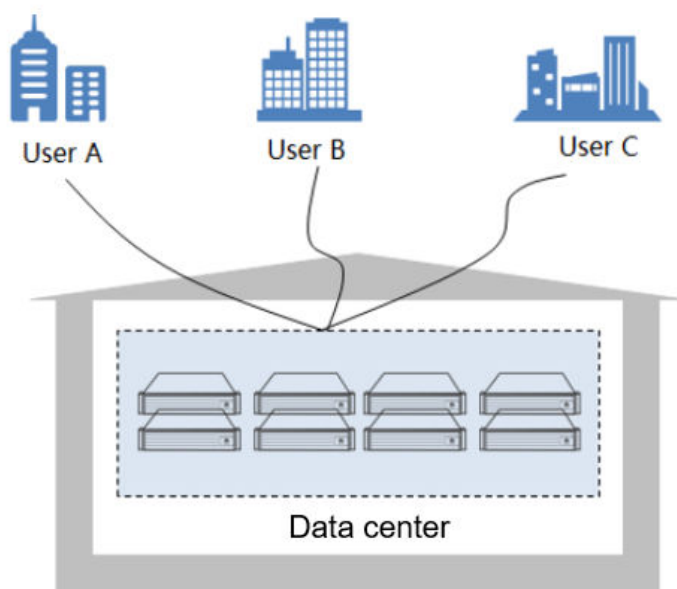
The type can be shared or exclusive.

4 Value-added Feature: Efficiency Improvement

- 4.1 SmartMulti-Tenant (Multi-tenancy)
- 4.2 SmartQuota (Quota)
- 4.3 SmartTier (Tiered Storage)
- 4.4 SmartEqualizer (Load Balancing)
- 4.5 SmartIndexing (Metadata Indexing)
- 4.6 SmartTakeover (Intelligent Takeover)
- 4.7 SmartQoS (Intelligent Quality of Service Control)
- 4.8 SmartAuditlog (Audit Log)
- 4.9 SmartEncryption (Data Encryption)
- 4.10 SmartDedupe and SmartCompression (Deduplication and Compression)
- 4.11 SmartMove (Online Volume Migration)
- 4.12 VMware Virtual Volume (vVol)
- 4.13 Scenario-specific SmartCompression (Scenario-specific Compression)
- 4.14 Standard SmartCompression (Standard Compression)
- 4.15 SmartMigration (Intelligent Data Migration)
- 4.16 SmartSync (Copying Objects to the Cloud)

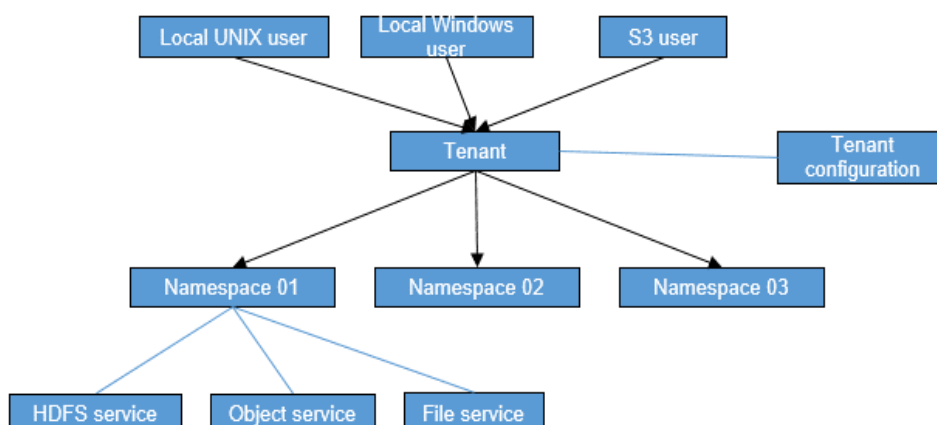
4.1 SmartMulti-Tenant (Multi-tenancy)

Figure 4-1 Multi-tenancy



As shown in the preceding figure, unstructured services of OceanStor Pacific series provide the multi-tenant management capability. Data of different tenants is logically isolated to facilitate resource division. The benefits are as follows: One system is managed in a centralized manner, resources are shared by multiple tenants, reducing initial investment, and logical data isolation improves tenant security.

Figure 4-2 Tenant management model



A tenant can have multiple namespaces. Each namespace can provide HDFS, object, and file services at the same time. One namespace can belong to only one tenant. SmartMulti-Tenant provides tenant-based unified resource allocation and

management. Multiple tenants share the same physical storage system. Resources of different tenants are isolated to ensure security and privacy. SmartMulti-Tenant provides a common and scalable multi-tenant management model, by which tenant-level configuration can be implemented, as described in the following table.

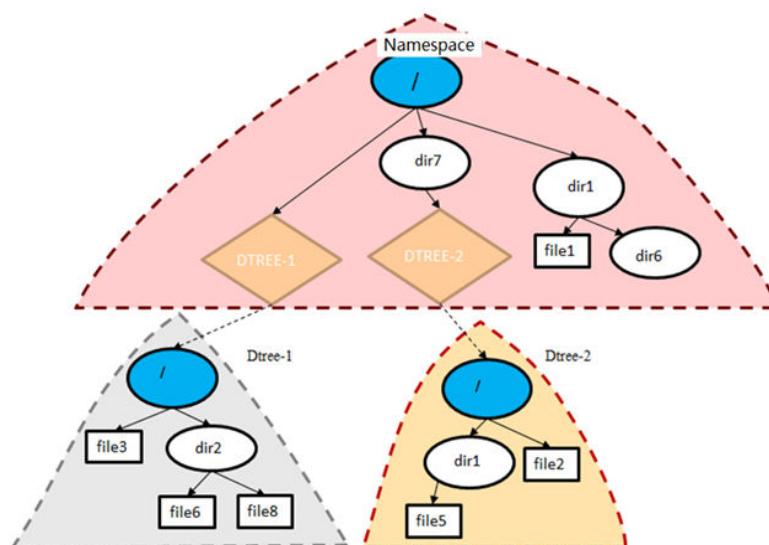
Table 4-1 Tenant-level configuration

Tenant-Level Configuration	Description
User management	Supports AD user, LDAP user, and NIS user management.
SmartQoS configuration	Supports tenant-level SmartQoS configuration.
Network management	Supports tenant-level subnet configuration.
SmartEncryption	Supports tenant-level SmartEncryption configuration.
SmartAuditlog	Supports tenant-level SmartAuditlog configuration.
SmartIndexing	Supports tenant-level SmartIndexing configuration.

4.2 SmartQuota (Quota)

SmartQuota is the quota management feature that limits or collects statistics of used storage space and file quantity. Unstructured services of OceanStor Pacific series provide the SmartQuota feature to apply to accounts, namespaces, or directory trees (dtrees). Administrators can configure proper quota policies to limit the usage of resources in accounts, namespaces, or dtrees. An alarm is sent when the resource usage reaches the configured threshold.

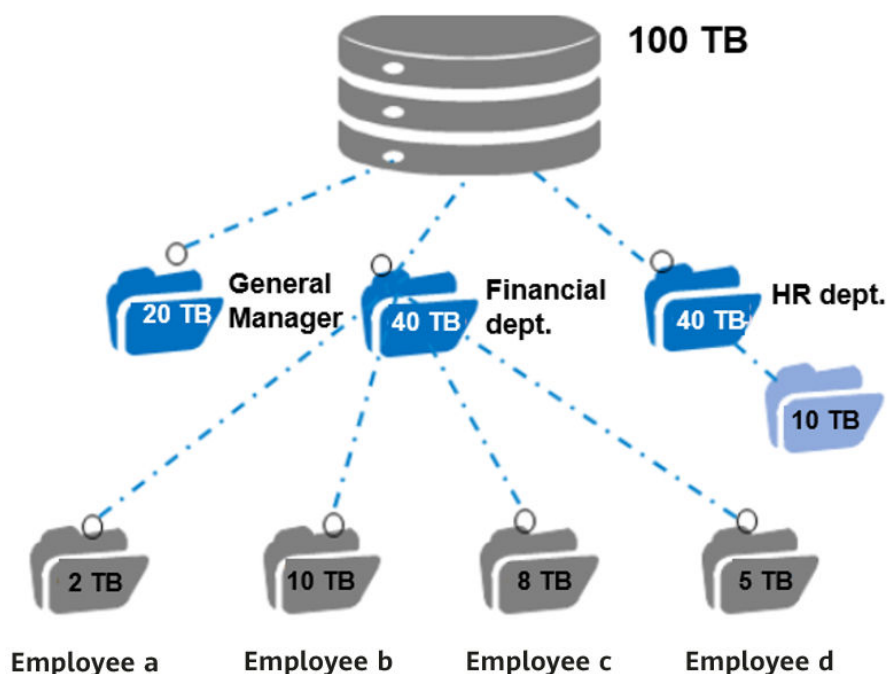
Figure 4-3 Dtree



Dtrees can be created in any directory, including the root directory of the file system, but dtrees cannot be nested.

SmartQuota enables administrators to properly allocate resources to accounts, users, user groups, or dtrees and prevent one account, user, user group, or dtree from occupying excessive storage resources. This avoids system breakdowns caused by uncontrolled storage resource usage and improves system security.

Figure 4-4 SmartQuota diagram



A file read, write, or creation operation delivered from the application layer will be forwarded to the quota module first. The quota module identifies the quota control items affected by the write operation, compares the items with quota

thresholds one by one, and determines whether to allow or forbid the operation based on the threshold comparison result. The system will continue to process the write operation and update statistics of the affected quota control items if none of the affected quota control items exceeds quota thresholds after this operation. If any item will exceed its threshold after this operation, the system forbids the operation immediately.

Quotas are classified into account quotas, user quotas, user group quotas, and directory quotas. You can specify the accounts, users, user groups, or dtrees to be controlled by a quota.

- **User quota**
A user quota controls the space usage or file quantity of users of the specified namespace or dtree.
- **User group quota**
A user group quota controls the space usage or file quantity of user groups of the specified namespace or dtree.
- **Directory quota**
A directory quota controls the root of a dtree or namespace. It limits the quota usage of all users. The directory quotas of a namespace and a dtree can be nested. With quota nesting, the system controls user resources to ensure that related quota items do not exceed the limit.
- **Account quota**
An account quota controls the total usage of all namespaces under an account.

Quota thresholds are classified into hard thresholds, soft thresholds, and recommended thresholds.

Table 4-2 Quota threshold description

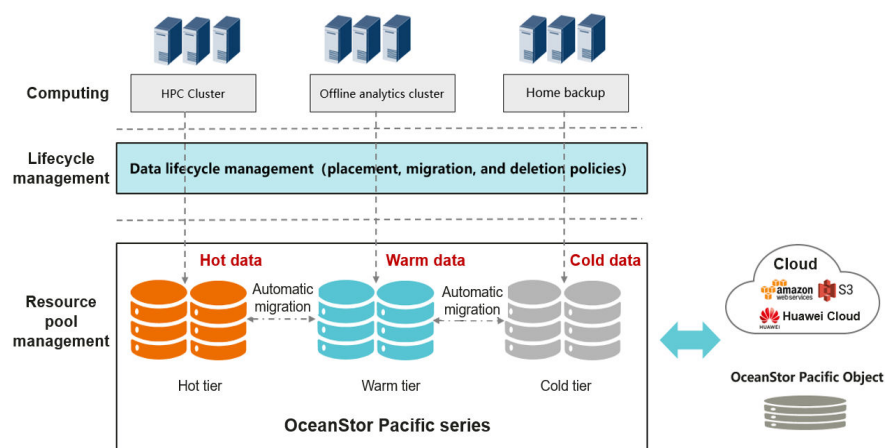
Threshold Type	Description
Recommended threshold	If the storage quota reaches the recommended threshold, the system generates an alarm but still allows writes.
Soft threshold	If the storage quota reaches the soft threshold, the system generates an alarm but still allows writes within the grace period. After the grace period elapses, the system immediately forbids writes and generates an alarm. You can configure a grace period together with the soft threshold. If the grace period is not configured, when the soft threshold is exceeded, the system only generates an alarm but still allows writes.
Hard threshold	If the storage quota reaches the hard threshold, the system immediately forbids writes and generates an alarm.

4.3 SmartTier (Tiered Storage)

For enterprises' services and applications, not all the data is important. Among the data, some is frequently accessed, some is seldom accessed, and some even has not been accessed for several years. Statistics and analysis show that data value has its lifecycle. In general, new data is frequently accessed and has high value. As time goes by, the data is accessed less frequently and is not accessed at last. The data value decreases year by year. A large amount of data with low value occupies high-performance and reliable system resources and large storage space, causing a waste of resources. However, the data cannot be deleted due to related policies and data warehouse construction requirements. How to store infrequently accessed data becomes one of the data management questions that enterprises must answer. Automatic tiered storage technology migrates data and stores the data in the proper storage space, solving the preceding problems.

Unstructured services of OceanStor Pacific series provide the SmartTier feature and allow the system to divide different types of physical nodes in the same storage pool into different disk pools. A disk pool is a collection of nodes that have the same characteristics such as the physical type and access performance. In addition, a storage pool can interconnect with heterogeneous devices (cloud or OceanStor Pacific) that support object protocols outside the storage pool. SmartTier allows users to define the value of data in workflows based on file pool policies. High-value files are stored on storage devices of high availability and high performance, and low-value files are stored on those of low cost, low performance, and low availability.

Figure 4-5 SmartTier diagram



As shown in the preceding figure, a tier is a collection of disk pools with the same features or capabilities in an OceanStor Pacific storage pool. A storage pool can interconnect with heterogeneous devices outside the storage pool. The storage space of OceanStor Pacific is classified into three tiers: hot, warm, and cold. Each disk pool corresponds to a storage tier. For example, the storage tier of an SSD node disk pool is hot, that of a SAS node disk pool is warm, and that of a SATA or NL-SAS node disk pool is cold. Each tier can contain multiple disk pools. The system automatically balances loads among multiple disk pools in a storage tier, including load balancing and capacity balancing. OceanStor Pacific can

interconnect with a heterogeneous device which functions as a storage pool with lower costs to store archive data that is not used for a long time.

SmartTier allows administrators to set file pool policies based on file properties. In this way, data can be automatically migrated between different disk pools (logical resource pools with EC data distribution or EC data fault isolation domains) or heterogeneous devices based on storage policies, ensuring high performance and proper use of the storage space. Users can manually plan their own services and store important directories and files in high-performance tiers based on their service characteristics. These policy conditions include the file name, file size, creation time, modification time, access time, status change time, and UID/GID. For objects, there are some unique attributes, such as prefix matching, object tag matching, expiration time matching of the current version, and expiration time matching of historical versions.

SmartTier supports intelligent tiering. This feature has a built-in machine learning (ML) module and supports seven automatic tiering dimensions: workload popularity, migration time, disk watermark, tier up/down, file/directory, file size, and capacity of each tier. Automatic and intelligent tiered data flow can be implemented without setting specific tiering conditions by administrators.

SmartTier provides placement, migration, and deletion policies.

- **Placement policy:**

A policy used to determine the initial location of file data. The policy conditions include file system/dtree, file name, and UID/GID. If no policy is matched, the default placement policy determines the placement location. Data can be stored only in an OceanStor Pacific storage pool.

- **Migration policy:**

- **Periodical migration policy:** determines the target location of file data migration. The policy conditions include the file system/dtree, file name, file size, creation time, modification time, access time, status change time, and UID/GID. The system automatically migrates data to a disk pool of the specified tier based on the policy. The data migration time can be set to prevent peak hours. By default, data migration is implemented at 00:00 every day. Data can be migrated to internal disk pools or heterogeneous devices.
- **One-off migration policy:** determines the target location of file data migration. The policy conditions include the file name, file size, creation time, modification time, access time, status change time, and UID/GID. The system automatically migrates data to a disk pool of the specified tier based on the policy. A one-off migration policy is executed only once when files are matched with it. Data can be migrated to internal disk pools or heterogeneous devices. In addition, data can be retrieved from heterogeneous devices to an OceanStor Pacific storage pool.

- **Deletion policy:**

- **Periodical deletion policy:** periodically deletes specified files. The policy conditions include the file name, file size, creation time, modification time, access time, status change time, and UID/GID. For a namespace with the S3 service enabled, you can specify the prefix, tag, expiration policy, and expiration days of non-current versions. The system automatically deletes files based on the deletion policy without any manual operations.

- **One-off deletion policy:** allows users to delete specified files at a time. Different from the periodical deletion policy, this task is executed only once.

 NOTE

All nodes in a disk pool must be of the same physical type. Otherwise, the performance of the disk pool is affected. The system collects statistics on disk space usage of a disk pool. If the occupied disk space exceeds the preset high watermark, new files cannot be saved or migrated to the disk pool.

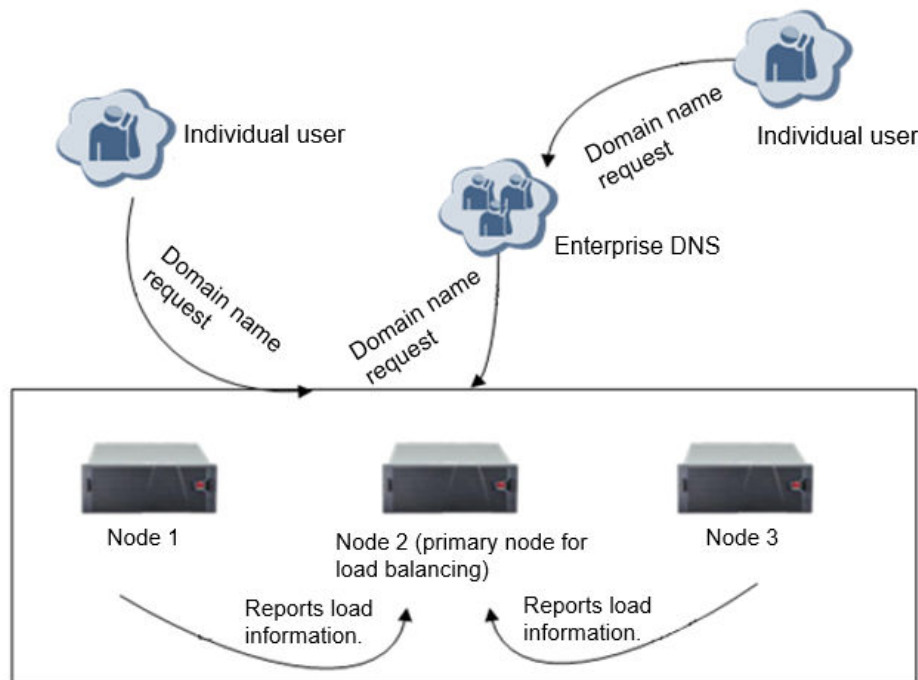
The system supports a maximum of 16 disk pools. Each disk pool must contain at least three physical nodes. Otherwise, the disk pool is unavailable. If only one disk pool exists in the system, you are not advised to use the SmartTier function.

Only storage devices that provide the standard AWS S3 protocol can be used as heterogeneous devices.

4.4 SmartEqualizer (Load Balancing)

The Huawei scale-out storage system provides the SmartEqualizer feature to implement load balancing for the file service. SmartEqualizer periodically collects information such as the CPU usage and network throughput of each node in the storage system and provides load balancing with user-definable balancing policies.

Figure 4-6 SmartEqualizer diagram



The SmartEqualizer service of OceanStor Pacific series involves one primary node and multiple secondary nodes. When a storage cluster is created, a node in the cluster is elected using the Paxos algorithm as the primary node for SmartEqualizer. The storage cluster has only one primary SmartEqualizer node at any time. If the primary node is faulty, the system automatically selects a new primary node. Each node periodically collects its load information, including the number of CPU cores, CPU frequency, memory size, network and NIC information,

CPU usage, memory usage, network throughput, and the number of NAS client connections. Then the node sends the collected load information to the elected primary node. The primary node uses the collected load information as the basis for load balancing. When receiving a new NAS mounting request, the storage cluster selects a proper node to process the request based on the load balancing policy configured by users.

The storage system provides unified domain name access externally, and the domain name query request service is integrated into the SmartEqualizer service. When a client initiates a domain name query request, SmartEqualizer calculates the load based on the current load balancing policy and returns the IP addresses of the proper nodes for the client to access the storage system.

The load balancing service of OceanStor Pacific series supports the following load balancing policies, which can be configured based on actual requirements:

- Round-robin mode (default policy): The system selects nodes in sequence to process client connection requests.
- CPU usage: The system selects the node with the lowest CPU usage to process client access requests.
- Number of node connections: The system selects the node with the least file service connections to process client connection requests.
- Node throughput: The system selects the node with the lowest network throughput to process client connection requests.

If a client does not have a DNS server, the client can set the DNS IP address of the storage system to the local DNS server address to directly access the DNS service of the storage system. If the client has its own DNS server, the client can configure domain name forwarding on the server to forward domain name access requests to the DNS service of the storage system.

NOTE

SmartEqualizer provides the domain name service only for HDFS and NAS services. The object service has an independent DNS for domain name management.

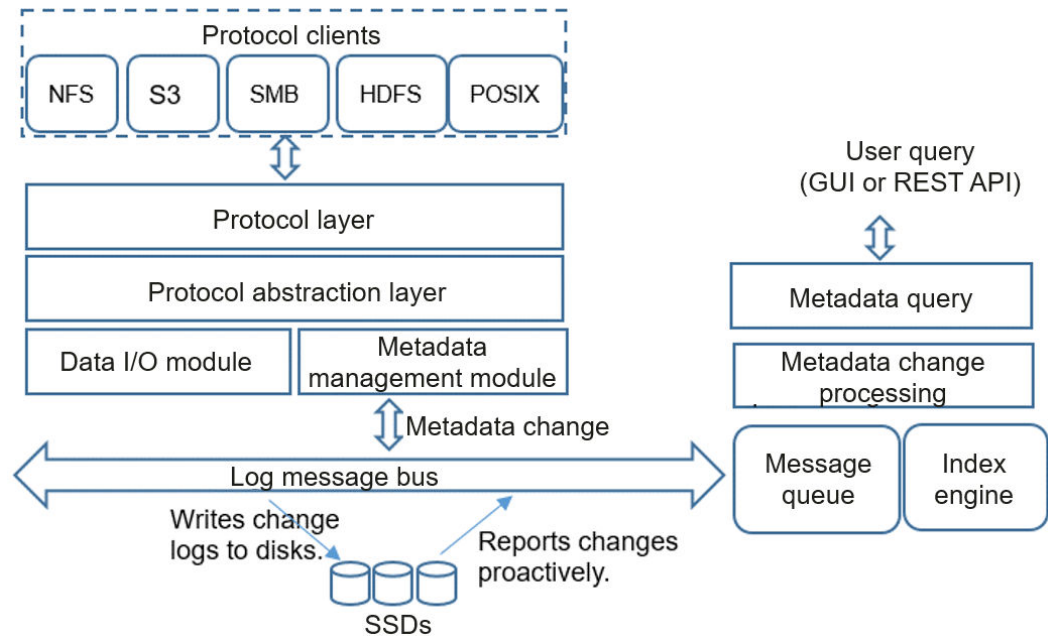
4.5 SmartIndexing (Metadata Indexing)

With the continuous development of information technologies and an ever-increasing amount of data, people pay more attention to mass data management. In mass data scenarios, users can manage mass data based on metadata. For example, users can obtain a list of image files with name extension **.jpeg**, a list of files with the size of greater than 10 GB, or a list of files created before a specific date. A simple metadata indexing operation refers to the process of searching for target files by running the **find** command on Linux. However, this method is too inefficient to meet customer requirements in scenarios where there are a large number of files, the directory hierarchy is deep, or search criteria are complex.

Unstructured services of OceanStor Pacific series provide the metadata indexing function based on unstructured storage pools. The metadata indexing function can be enabled or disabled by namespace (or file system). For file systems where metadata indexing is enabled, after front-end service I/Os change the metadata of files, objects, or directories, the changed metadata is proactively pushed to the index system in asynchronous mode. Different from periodic metadata scanning

used by traditional storage products, metadata indexing prevents the impact on the performance of the production system and meets users' requirements for quick search of mass data.

Figure 4-7 Metadata indexing principle

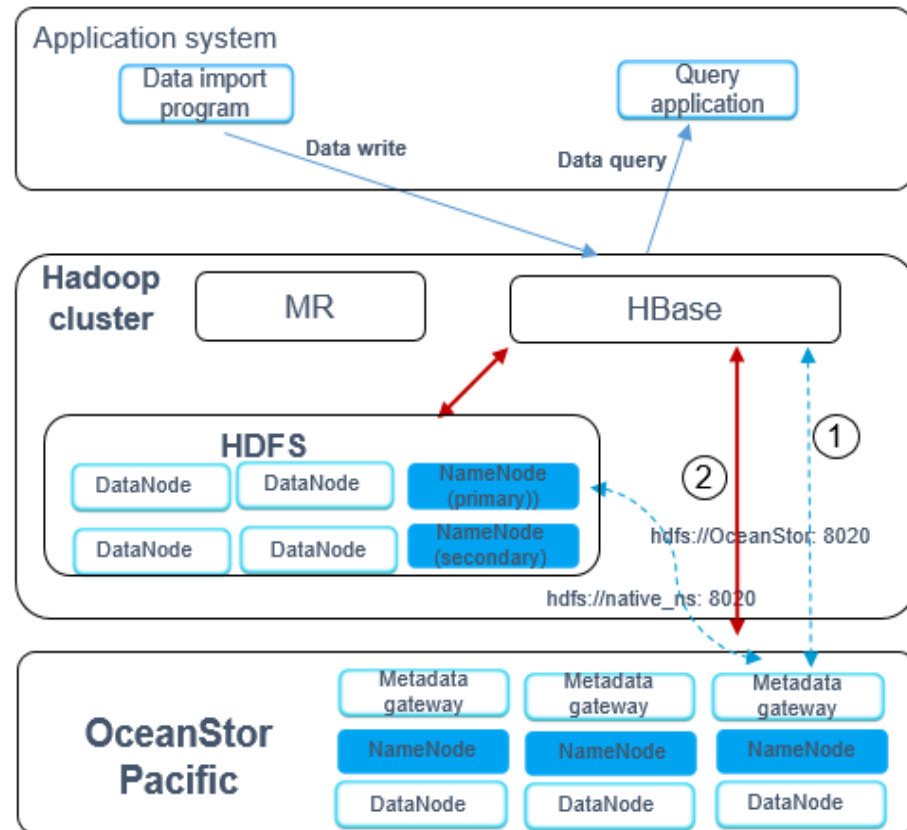


Metadata change operations (such as file creation and file permission, user, user group, and file data modification) from different protocols are forwarded to the owning node of the corresponding file or directory and processed by the metadata management module on the node. In addition to metadata change persistence, unstructured services of OceanStor Pacific series introduce fast transaction processing and write metadata change logs and metadata after changes to acceleration disks. In this way, metadata changes and metadata change logs will be both processed successfully or fail to be processed, ensuring metadata consistency. After being written to disks, the change logs are proactively reported to the message queue for metadata indexing through the log message bus. Then, they are read by the metadata change processing module and the indexes are updated in the index engine.

4.6 SmartTakeover (Intelligent Takeover)

The OceanStor Pacific series HDFS service provides the SmartTakeover metadata gateway solution and global namespace to manage other HDFS namespaces without migrating data. Seamless access can be implemented on the existing HDFS platform without affecting upper-layer services.

Figure 4-8 Principle of the HDFS metadata gateway



Modifying the configuration:

1. In the Hadoop cluster, change the data access path of HBase/Hive/Spark/Presto/openLookeng to a NameNode of the OceanStor Pacific series HDFS service.
2. Add the NameNode path of the original HDFS platform to the metadata gateway.

Data access paths:

1. HBase/Hive/Spark/Presto/openLookeng queries the data location through the metadata gateway in path 1. If the data is not in the HDFS storage pool, the data location on the original HDFS platform is obtained through path 2 and the location information is returned to the components.
2. The components obtain data based on the location path.

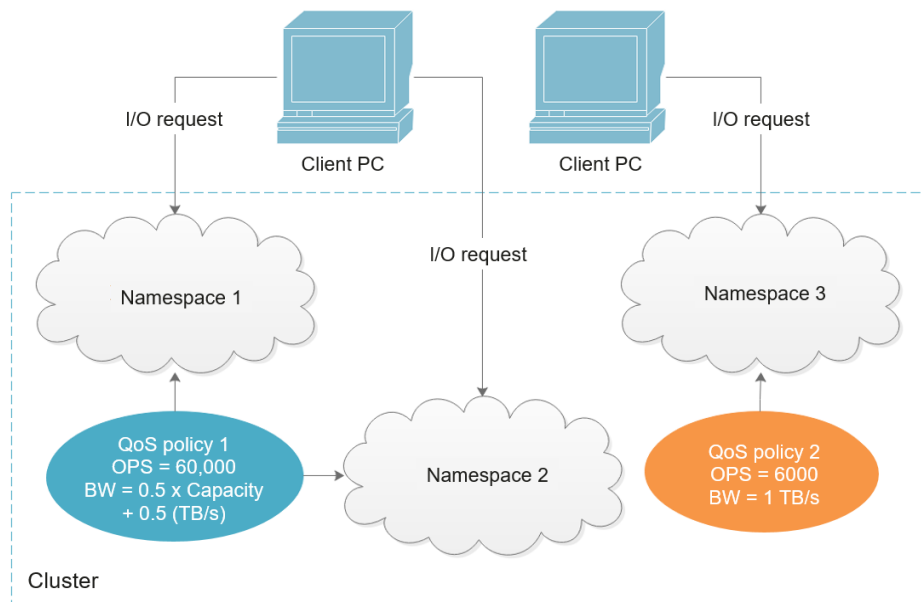
4.7 SmartQoS (Intelligent Quality of Service Control)

4.7.1 Unstructured Data Storage Services

Unstructured services of OceanStor Pacific series provide the SmartQoS technology to configure QoS policies by tenant, namespace, client, user, or directory.

Namespace-level QoS policies can be configured based on the used capacity or preset upper limit (bandwidth and IOPS) to flexibly meet requirements in different scenarios. As shown in the following figure, **Namespace 1** and **Namespace 2** are associated with **QoS policy 1**. The maximum OPS of each namespace is limited to 60,000, and the maximum bandwidth is calculated based on the used capacity of each namespace. **Namespace 3** is associated with **QoS policy 2**. Its maximum OPS is limited to 6000, and its maximum bandwidth is limited to 1 TB/s. You can separately set the OPS and bandwidth upper limits for read and write operations in namespaces of the object service.

Figure 4-9 Namespace-based QoS policy



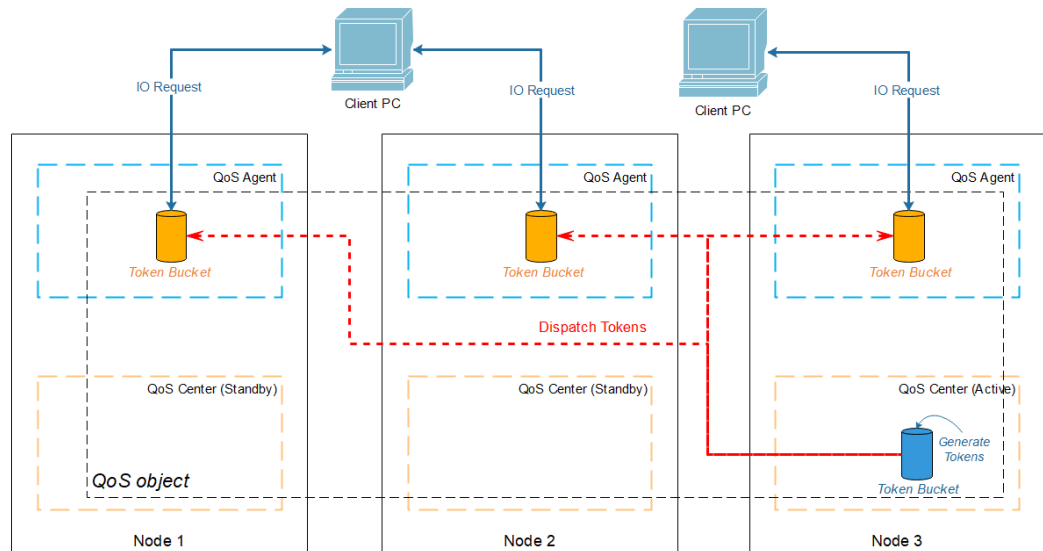
Tenant-level QoS supports the upper limit control of OPS and bandwidth. All namespaces of a tenant share QoS. The tenant-level QoS control and namespace-level QoS control use the overlay control principle. That is, the OPS upper limit and bandwidth upper limit of any namespace must meet the QoS policy rules configured for the namespace and the tenant to which the namespace belongs.

Client-level QoS control supports the upper limit control of OPS and bandwidth based on client IP addresses. For Distributed Parallel Client (DPC), the IP address is the management IP address of DPC. For common clients, the IP address is the service IP address that can be configured by users. Similarly, according to the overlay control principle, in addition to the QoS policy configured, OPS and bandwidth upper limits of services delivered by any client IP address must meet the QoS policy rules of the namespace and the tenant to which the namespace belongs. OPS and bandwidth upper limits of read and write operations delivered by the client service IP address of the object service can be set separately.

User-level QoS allows you to set bandwidth and OPS upper limits by user. The object service supports user QoS, but you need to configure the mapping from S3 users to UNIX users. Similarly, according to the overlay control principle, in addition to the QoS policies configured for a user, the QoS policies configured for the namespace to which the user belongs, related tenants, and related clients must be met.

Directory-level QoS allows you to set bandwidth and OPS upper limits by directory. Similarly, according to the overlay control principle, in addition to the QoS policies configured for a directory, the QoS policies configured for the parent directory of the directory, namespace to which the directory belongs, related tenants, related users, and related clients must be met.

Figure 4-10 QoS technical principles of unstructured services



In the scale-out architecture, QoS Agent and QoS Center modules are installed on all nodes in a cluster. Some nodes are elected as active Center nodes, on which the QoS Center modules are in the active state, while the QoS Center modules on other Center nodes are in the standby state. The QoS Center modules on active Center nodes manage the QoS Agent modules on all nodes in groups. The QoS Center modules on active Center nodes manage the total number of tokens for QoS objects based on the upper limits of traffic control and concurrent resources. A client can access any node. An I/O sent to the namespace applies for tokens from the associated QoS object through the QoS Agent module. The I/O can be processed only after the tokens are obtained successfully.

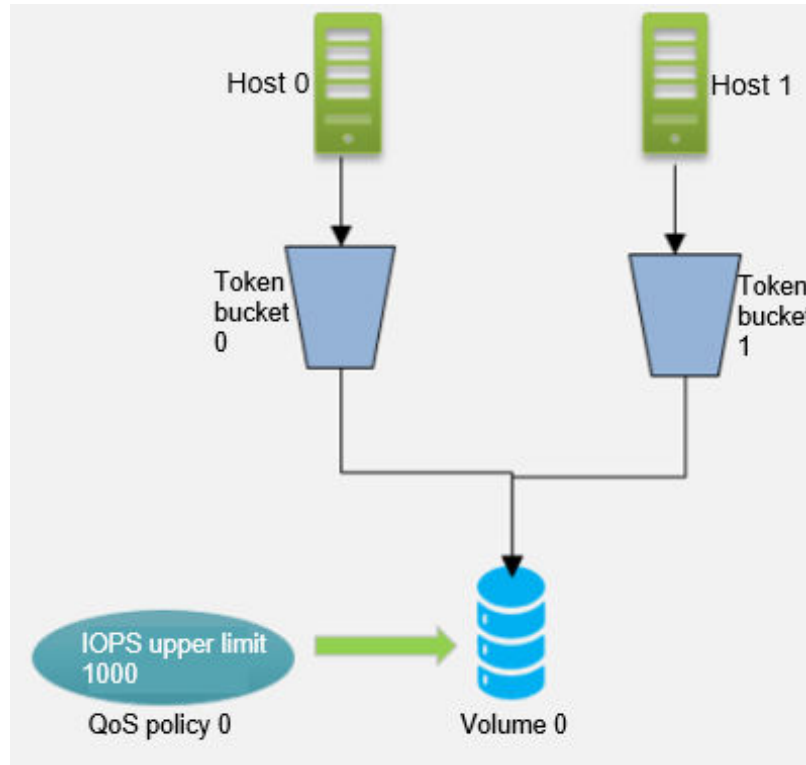
4.7.2 Structured Data Storage Service

The structured service of OceanStor Pacific series provides SmartQoS to set performance objectives for volumes and storage pools by specifying bandwidth and IOPS upper limits. The total read and write performance, read performance, and write performance can be limited. QoS policies can take effect in specified time ranges based on service loads to prevent I/O storms from affecting production services.

The structured service of OceanStor Pacific series provides the QoS function based on policy-based distributed traffic control parameter coordination and traffic control management. Traffic is limited based on the performance control objectives (IOPS and bandwidth) set by users. The I/O traffic control mechanism prevents some services from affecting other services due to heavy traffic. In addition, quotas that exceed the benchmark performance can be used within a certain period of time. The following describes the QoS traffic control technology of OceanStor Pacific series.

- Self-adaptive adjustment algorithm based on negative feedback
When a volume is mounted to multiple VBS nodes in OceanStor Pacific series, the system resources consumed by services on the volume need to be controlled. That is, the overall performance of the volume needs to be limited. This requires coordination of distributed traffic control parameters.

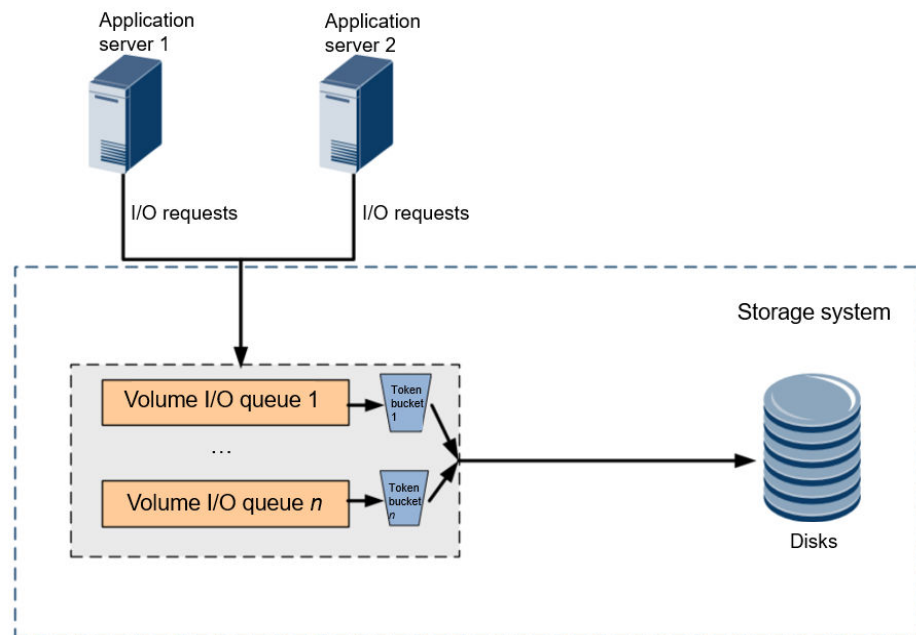
Figure 4-11 Self-adaptive adjustment algorithm based on negative feedback



Suppose that the system is in the initial state and the IOPS upper limit of **Volume 0** is 1000, as shown in the preceding figure. The system will detect the service pressure from **Host 0** and **Host 1** on **Volume 0**, and adaptively adjust the speed of tokens in **Token bucket 0** and **Token bucket 1** to limit the maximum IOPS to 1000.

- Volume-based I/O traffic control management algorithm
QoS traffic control management is implemented by volume I/O queue management, token allocation, and dequeuing control. After you set a performance upper limit objective for a QoS policy, the system determines the performance upper limit of each VBS node by coordinating the distributed traffic control parameters and then converts the performance upper limit into a specified number of tokens. If the traffic to be restricted is IOPS, one I/O consumes one token. If the traffic to be restricted is bandwidth, one byte corresponds to one token. Volume-based I/O queue management allocates storage resources using the token mechanism. The more tokens a volume has, the more I/O resources the system allocates to the volume.

Figure 4-12 Volume-based I/O traffic control management



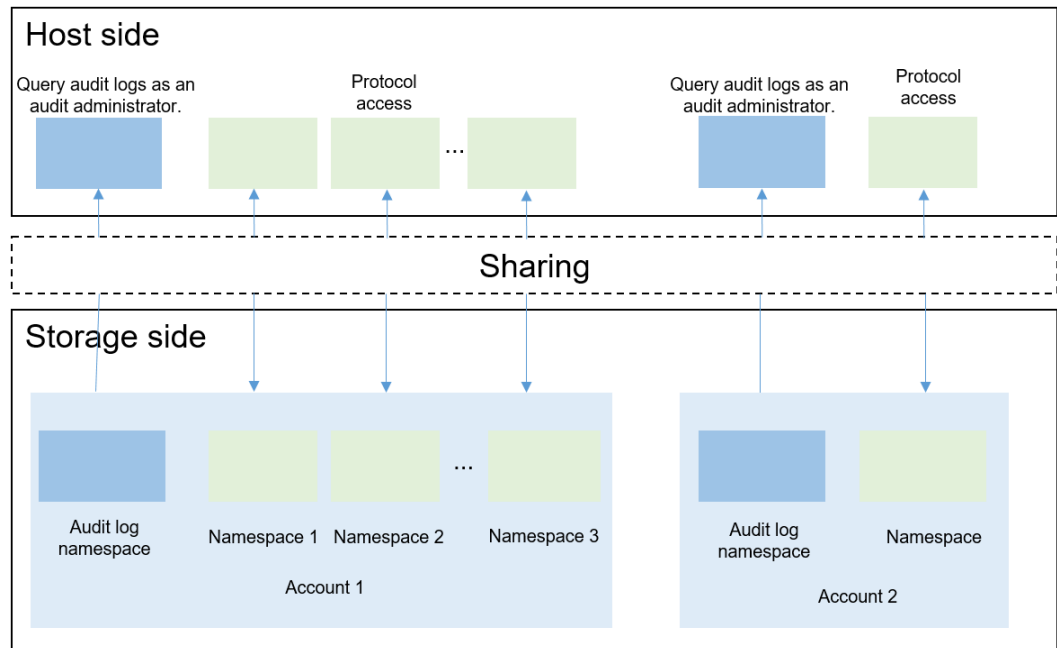
As shown in the preceding figure, I/Os of applications first enter their I/O queues after entering the storage system. SmartQoS periodically processes I/Os waiting in the queues. It dequeues the head element in a queue, and attempts to obtain tokens from a token bucket. If the number of remaining tokens in the token bucket meets the token requirement of the head element, the system delivers the element to another module for processing and continues to process the next head element. If the number of remaining tokens in the token bucket does not meet the token requirement of the head element, the system puts the head element back in the queue and stops I/O dequeuing.

4.8 SmartAuditlog (Audit Log)

Working principles

Unstructured services of OceanStor Pacific series support namespaces that can be accessed through multiple protocols. When multiple protocols are supported, users of NFS, SMB, object, and HDFS protocols can access specified files. With the SmartAuditlog feature of OceanStor Pacific series, you can configure the audit switch and operation words on a namespace to record file or directory operations on the namespace. Audit logs can be recorded in the audit log namespace. The log file format can be EVT, XML, or TXT.

Figure 4-13 SmartAuditlog feature diagram



1. When the protocol layer of scale-out storage nodes receives file access operations from protocol clients (NFS, SMB, object, HDFS, and FTP), the protocol layer determines whether to record audit logs and assembles log information. If logs are recorded, the protocol layer sends the assembled audit log information to the audit log module.
2. The audit log module writes the log records to a temporary storage area.
3. A background audit log task collects the logs in the temporary storage area and stores the logs in the audit log namespace in EVT, XML, or TXT format according to the account configuration requirements.
4. After the audit log format is converted, delete the logs in the temporary storage area.
5. Only one read-only audit log namespace can be configured for each account. You can configure an NFS or CIFS share to access log files in the audit log namespace.

Audit log items

OceanStor Pacific series supports the following file access audit operations: **Create, Delete, Rename, Read, Write, Open, Close, List folders, Obtain properties, Set properties, Obtain security properties, Set security properties, Obtain extension properties, Set extension properties, and Concatenate files.**

OceanStor Pacific series supports the following user audit operations: **Login and Logout.**

Table 4-3 Operations supported by audit logs

Event Name	Description	Example
Create	A file or directory was created.	Open(creat flag), mkdir, put a object
Rename	A file or directory was renamed.	Rename, mv
Delete	A file or directory was deleted.	Unlink, rmdir, rm
Read	A file was read.	cat
Write	A file was written.	cp, hdfs dfs -put
Open	A file was opened for reading or writing.	cat, cp
Close	A file was closed.	close, cp
List folders	A directory was listed.	ls, ll, getListing
Obtain properties	A file's or directory's attributes were obtained.	stat
Set properties	A file's or directory's attributes were set.	chmod, chown, hdfs dfs -chmod
Obtain security properties	A file's or directory's security attributes were obtained.	nfsv4acl-nfs4_getfacl, hdfs dfs -getfacl
Set security properties	A file's or directory's security attributes were set.	nfsv4acl-nfs4_setfacl, modifyacl
Obtain extension properties	A file's or directory's extended attributes were obtained.	hdfs dfs -getfattr
Set extension properties	A file's or directory's extended attributes were set.	hdfs dfs -setfattr
Concatenate files	Files were concatenated.	concate
Login	User login.	An FTP or CIFS user logs in to a shared directory.
Logout	User logout.	An FTP or CIFS user logs out.

Information recorded in audit logs

The following table lists the fields supported in multi-protocol audit logs.

Table 4-4 Key fields in audit logs

Event Information	Event Field	Description
System	Provider Name	Name of the event audit provider
	EventID	Event ID
	Version	Event version
	EventName	Event name
	Level	Event severity
	Source	Event protocol type
	Result	Event result
	Opcode	Event operation
	Keywords	Event keyword
	TimeCreated System Time	Event occurrence time
	Channel	Event occurrence channel
	Computer	Computer where the event occurs
EventData	SubjectIP	Event subject IP address
	SubjectHostName	Event subject host name
	SubjectUnix	Event subject UNIX attribute
	SubjectUserSid	Event subject user SID
	SubjectUserName	Event subject user name
	SubjectDomainName	Event subject domain name
	SubjectUserIsLocal	Whether the event subject user is a local user
	ObjectServer	Event object server
	ObjectType	Event object type
	ObjectName	Event object name
	HandleID	Event object ID
	OldDirHandle	Old event directory ID NOTE This field is valid only when operation event rename is generated.
	NewDirHandle	New event directory ID NOTE This field is valid only when operation event rename is generated.

Event Information	Event Field	Description
	OldPath	Old event object path NOTE This field is valid only when operation event rename is generated.
	NewPath	New event object path NOTE This field is valid only when operation event rename is generated.
	AccessList	Access permission list NOTE This field is valid only when operation event creation, open, or soft link is generated.
	AccessMask	Access mask NOTE This field is valid only when operation event creation, open, or soft link is generated.
	DesiredAccess	Required access permission NOTE This field is valid only when operation event creation, open, or soft link is generated.
	Attributes	Attribute NOTE This field is valid only when operation event creation, open, rename, or soft link is generated.
	LinkName	Link name NOTE This field is valid when a hard link is created.
	InformationRequested	Request information list NOTE This field is valid only when the operation event of listing folders, obtaining attributes, or obtaining security attributes is generated.
	InformationSet	Information list NOTE This field is valid only when the operation event of setting attributes is generated.

Event Information	Event Field	Description
	WriteOffset	Write offset NOTE This field is valid only when a write operation event is generated.
	WriteCount	Write length NOTE This field is valid only when a write operation event is generated.
	ReadOffset	Read offset NOTE This field is valid only when a read operation event is generated.
	ReadCount	Read length NOTE This field is valid only when a read operation event is generated.
	oldSD	Old security description NOTE This field is valid only when the operation event of setting security attributes is generated.
	newSD	New security description NOTE This field is valid only when the operation event of setting security attributes is generated.
	SearchPattern	Search mode NOTE This field is valid only when the operation event of listing folders is generated.
	SearchFilter	Searching and filtering NOTE This field is valid only when the operation event of listing folders is generated.
	NameList	Name list of the source files involved in the CONCAT operation NOTE The length of NameList is limited. When the total length of the name list of the source files in the CONCAT operation exceeds the upper limit, the excessive part of the source file names will not appear in NameList .

Event Information	Event Field	Description
	FileCountInList	Number of files recorded in the current name list NOTE This field is valid only when a CONCAT operation event is generated.

4.9 SmartEncryption (Data Encryption)

OceanStor Pacific series supports two types of static data encryption: software and hardware-integrated encryption and self-encrypting disk (SED), protecting data stored in the storage system from being disclosed.

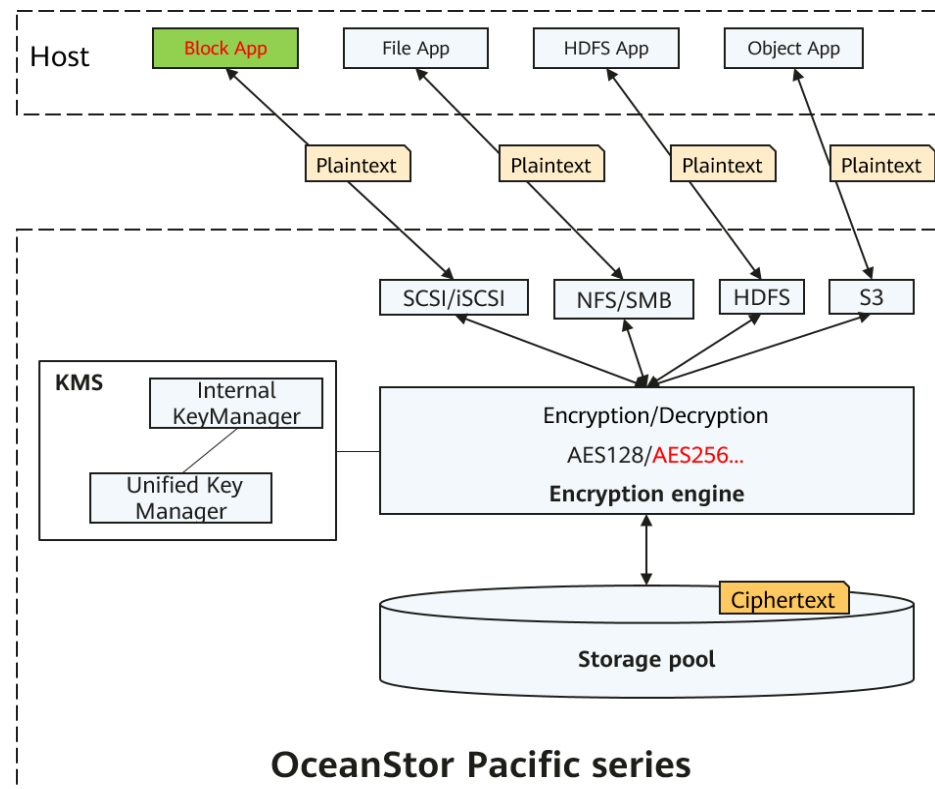
- Software and hardware-integrated encryption
Both structured and unstructured data supports software-based encryption. Three encryption algorithms are supported: XTS-AES-128, XTS-AES-256, and SM4-XTS. Theoretically, compared with XTS-AES-128, XTS-AES-256 has a no more than 10% impact on performance. Software and hardware-integrated encryption encrypts static data using an Internal Key Manager and a built-in encryption engine and provides account-level security through two-layer key management. After data is written to a storage system, the data is encrypted using software and hardware and then written to disks for storage. When the data is read, the ciphertext data is read from the disks, decrypted using software and hardware to plaintext, and then provided to users.
- SED
SEDs can be configured for structured and unstructured services to encrypt data on disks. SEDs provide two-layer security protection by using an authentication key (AK) and a data encryption key (DEK).

4.9.1 Software and Hardware-integrated Encryption

Software and hardware-integrated encryption adopts an Internal Key Manager and a built-in encryption engine to encrypt static data and provides tenant-level security through two-layer key management, ensuring data encryption persistence and key management security. The built-in encryption engine provides basic capabilities for data encryption and decryption and supports the AES-XTS algorithm. In addition, supports ASE-NI instruction acceleration to improve performance when data is encrypted and decrypted using software. Two-layer key management refers to allocating authentication keys (AKs) by tenant. Data encryption keys (DEKs), used to encrypt namespace data of tenants, are protected by the AKs of the corresponding tenants.

If you want to enable encryption for a namespace or volume, you need to enable encryption for the tenant of the namespace or volume. After encryption is enabled, the tenant identity authentication key AK will be applied for from the Internal Key Manager through the encryption channel. Then, create an encrypted namespace or volume for the tenant and apply for a DEK. The DEK is encrypted using the AK of the tenant.

Figure 4-14 Encryption principles



As shown in the preceding figure, encrypted data of unstructured services can be accessed through multiple protocols. For example, a file application client writes plaintext data through the NFS or SMB protocol, obtains the key from the Key Manager at the encryption engine layer, encrypts the data, and then stores data in the storage pool persistently. When data is read using other protocols such as HDFS and S3, the encryption engine layer reads the ciphertext data from the storage pool, obtains the key from Internal Key Manager, decrypts the data, and returns the plaintext data to the protocol layer. The protocol layer returns the plaintext data to the client application.

NOTE

Data encryption cannot be disabled once it is enabled.
The SM algorithm requires an independent license.

4.9.2 SED

The SEDs, Internal Key Manager, and storage system work together to encrypt static data, ensuring data security.

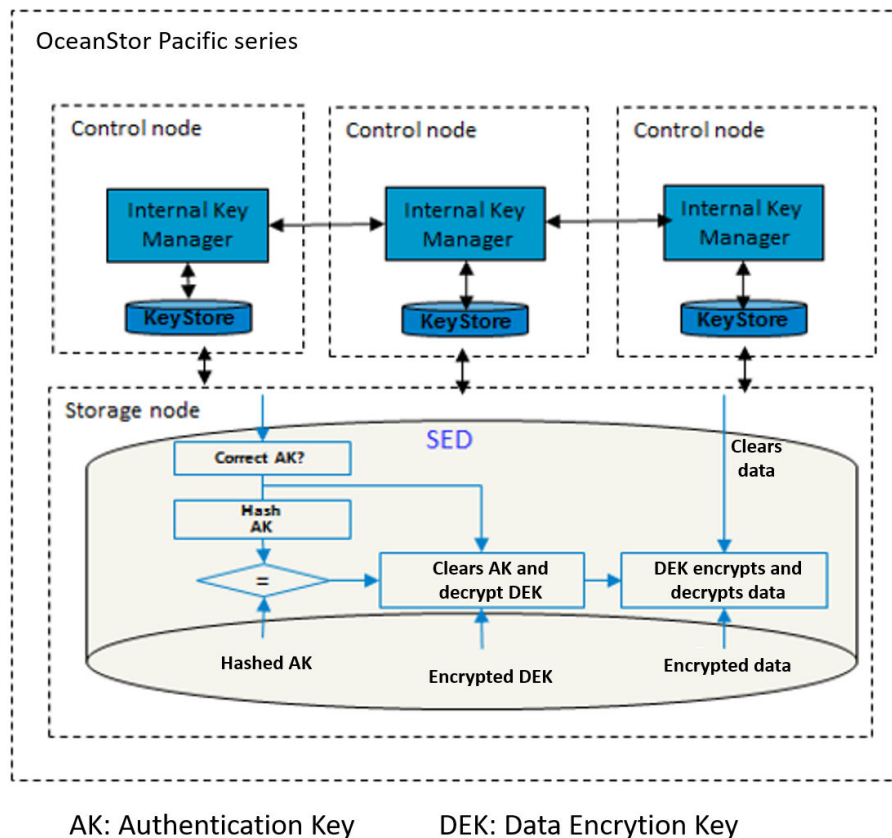
Data encryption brings the following benefits:

- By using SEDs, data is encrypted and decrypted on disks, eliminating impacts on service application processing.
- Data is encrypted after being flushed to disks without affecting upper-layer value-added features, such as data deduplication and compression.
- Fast data destruction is supported by destroying keys.

- Internal Key Manager (a key management application embedded in storage systems) is easy to deploy, configure, and manage.

OceanStor Pacific series storage systems adopt a scale-out architecture to manage Internal Key Manager. Internal Key Manager deployed on multiple nodes works together to provide secure key services for SEDs. The following figure shows the implementation.

Figure 4-15 Encryption principles of OceanStor Pacific series SEDs



- AK mechanism: When data encryption is enabled, the storage system enables AutoLock of SEDs and authenticates and controls the access to SEDs using AKs allocated by Internal Key Manager. In this case, the access is protected by AutoLock and only the storage system itself can access its SEDs. When the storage system accesses an SED, it acquires an AK from Internal Key Manager. If the AK is consistent with that of the SED, the SED decrypts the DEK for data encryption and decryption. If the AKs are inconsistent, read and write operations will fail.
- DEK mechanism: After AutoLock authentication is successful, SEDs use built-in encryption chips and internal DEKs to encrypt or decrypt the data that is written or read. When you write data, the system encrypts the plaintext data using the DEK of the AES encryption engine and then writes the data into storage media. When you read data, the system decrypts the requested data into plaintext using the DEK. The DEK cannot be acquired separately, which means that the original information on an SED cannot be read directly after the SED is removed from the storage system.

 NOTE

In actual deployment scenarios, multiple storage nodes are deployed as control nodes, and all storage nodes are deployed with services of storage nodes.

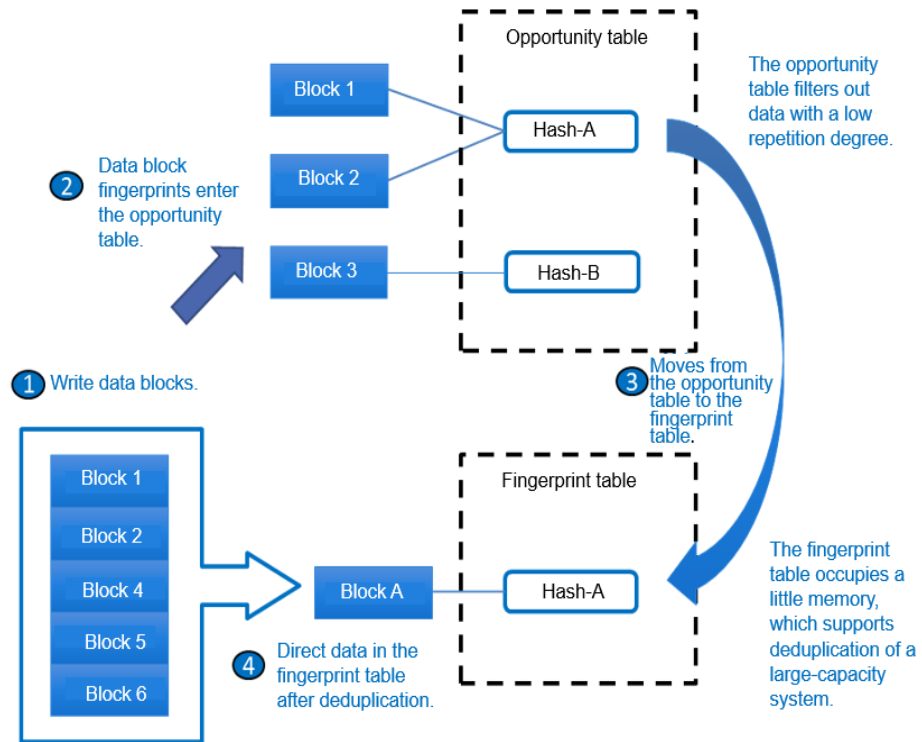
4.10 SmartDedupe and SmartCompression (Deduplication and Compression)

To cope with the increase in operation costs incurred by management data growth, storage device vendors introduce data reduction techniques, including data deduplication and compression, to reduce the amount of data that is actually stored. The OceanStor Pacific series structured service adopts Huawei-developed SmartDedupe and SmartCompression features to implement data reduction.

The OceanStor Pacific series structured service adopts intelligent self-adaptive deduplication technology, oriented to user requirements. When the service load is heavy, inline deduplication is disabled automatically to ensure performance and post-process deduplication is implemented to delete duplicate data. When the service load is light, inline deduplication is enabled automatically to prevent read/write amplification of post-process deduplication. The intelligent self-adaptive deduplication technology is user-oriented and automatically switches between inline and post-process deduplication based on service loads without user awareness, making the most of the two deduplication modes.

To achieve a better deduplication and compression effect, the OceanStor Pacific series structured service adopts global deduplication. The scale-out storage space is huge. To reduce the memory space consumed by the fingerprint table, an opportunity table is introduced, as shown in the following figure. The fingerprints of data blocks are first recorded in the opportunity table for counting. When the number of data blocks with a same fingerprint reaches a specific threshold (3 by default and modifiable), the system promotes the fingerprint to the fingerprint table and implements data deduplication accordingly.

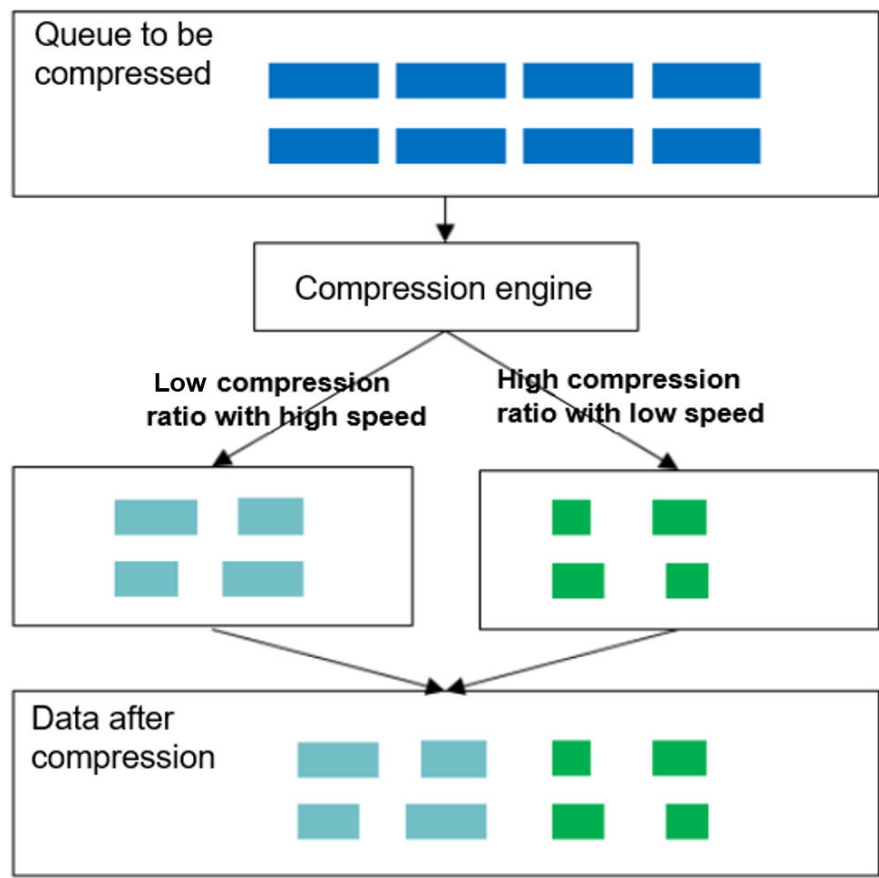
Figure 4-16 Deduplication process of OceanStor Pacific series



If compression is disabled, the system directly applies for storing the data based on the length of data blocks. If compression is enabled, the system first compresses data blocks and then data blocks are saved in the system at the granularity of 512 bytes.

The OceanStor Pacific series structured service provides a compression engine which adopts two different compression algorithms. One is the algorithm with low compression ratio but high speed, and the other is the algorithm with high compression ratio but low speed. By configuring different execution ratios of the two compression algorithms, you can obtain different performance and data reduction ratios. Only one compression algorithm can be configured for a storage pool. Changing the compression algorithm of a storage pool does not affect compressed data. During data reads, compressed data will be decompressed using the same compression algorithm when the data was compressed.

Figure 4-17 Multi-policy compression of the OceanStor Pacific series structured service



NOTE

After data encryption is enabled, the deduplication and compression efficiency becomes lower.

4.11 SmartMove (Online Volume Migration)

SmartMove developed by Huawei is also called intelligent cross-pool online migration. It is a key technology for service migration and can implement cross-pool online migration of service data in a storage system.

Table 4-5 describes the benefits of SmartMove.

Table 4-5 Benefits of SmartMove

Benefit	Description
Reliable service continuity	Service data is migrated online non-disruptively, preventing any loss caused by service interruption during service migration.

Benefit	Description
Stable data consistency	During service data migration, data changes made by hosts will be sent to both the source LUN and target LUN, ensuring data consistency after migration and preventing data loss.
Convenient performance adjustability	Based on service requirements, data can be migrated between different storage pools to flexibly adjust service performance.

The working principles of SmartMove are as follows:

SmartMove replicates all data from a source LUN to a target LUN and exchanges the data volumes of the source and target LUNs online after the replication is complete. Then, the target LUN completely takes over services from the source LUN.

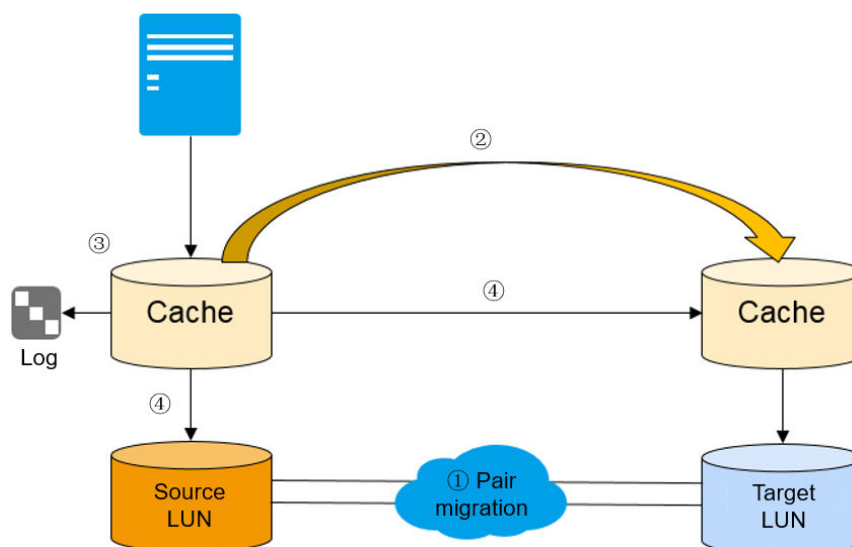
The implementation of LUN migration consists of the following two phases:

1. Service data synchronization

A storage system initiates a background task to synchronize data from the source LUN to the target LUN. After the synchronization is complete, service data on the target LUN is the same as that on the source LUN.

- ① Before migration, you must configure the source and target LUNs.
- ② When migration starts, the source LUN replicates data to the target LUN in the background.
- ③ During migration, the host can still access the source LUN. When the host writes data to the source LUN, the request is recorded in a log. The log only records the address information but no data content.
- ④ Data is written to both the source and target LUNs.
 - The system waits for the write response from the source and target LUNs. If data is successfully written to both LUNs, the log is cleared. Otherwise, the log is retained. In this case, the synchronization is interrupted abnormally. When the synchronization is started later, the data blocks corresponding to the log address are replicated again.
 - The system returns the write result of the source LUN to the host. For example, when the target LUN is faulty, the execution result of the write I/O request to the target LUN does not affect services on the source LUN.
- ⑤ The system performs the preceding operations until all data is replicated to the target LUN.

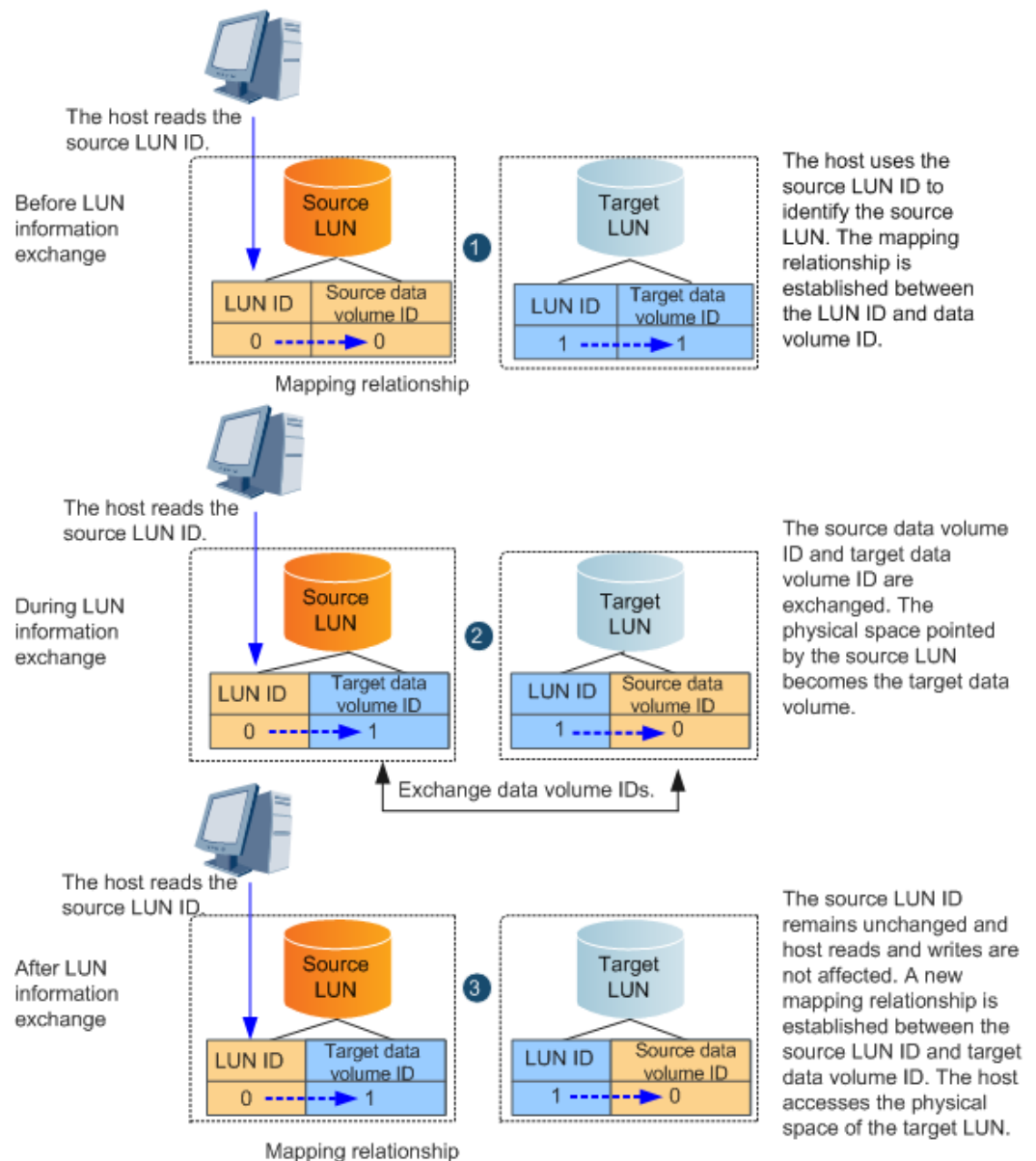
Figure 4-18 Synchronizing changed data during service migration



2. LUN information exchange

LUN information exchange is the prerequisite for the target LUN to take over services from the source LUN. LUN information exchange enables services on the source LUN to be migrated to the target LUN without affecting host services. In this way, the target LUN can completely replace the source LUN to run services. [Figure 4-19](#) shows the LUN information exchange process.

Figure 4-19 LUN information exchange process



① In a storage system, each LUN and its corresponding data volume have their unique identifiers, namely, LUN ID and data volume ID. A source LUN corresponds to a source data volume. The former is a logical concept whereas the latter is a physical concept. A host identifies a source LUN using the source LUN ID, and a source LUN identifies a source data volume using the source data volume ID.

② LUN information exchange changes the mapping relationship between a LUN ID and a data volume ID. While the source LUN ID and target LUN ID remain unchanged, the source data volume ID exchanges with the target data volume ID. As a result, the source LUN ID corresponds to the target data volume ID, and the target LUN ID corresponds to the source data volume ID. In addition to the data volume ID, LUN attributes, such as the owning storage pool ID and owning disk domain ID, are also exchanged.

③ As host services keep running, the mapping relationship between the host and the source LUN remains unchanged. Though the host still identifies the source LUN ID after service migration, the LUN information exchange enables the source LUN to point to the target data volume, making users unaware of the service migration.

4.12 VMware Virtual Volume (vVol)

vVol is a new function of VMware ESXi 6.0. It provides storage resources of a virtual machine (VM) and delivers more VM operations to the storage system, fully utilizing storage resources. The vVol technology lays a foundation for VM I/O isolation and storage's VM awareness.

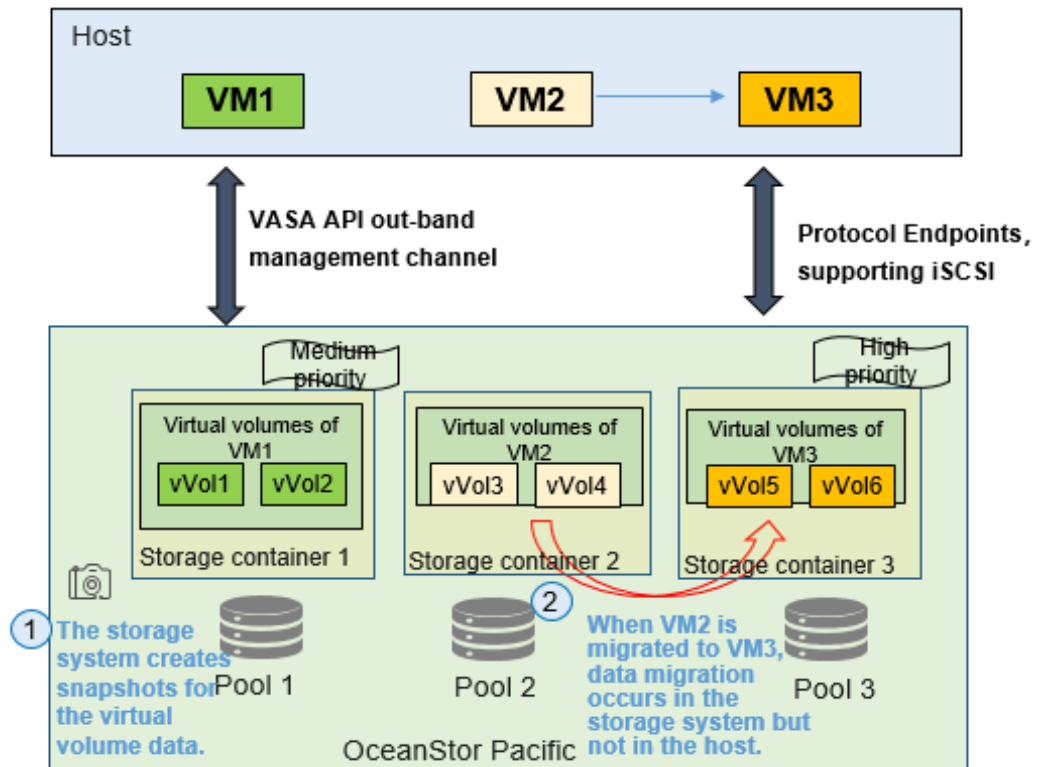
The OceanStor Pacific series structured service supports the vVol 1.0 technology which enhances the VMware ecosystem compatibility, simplifies host management, releases host resources, and improves efficiency. The vVol technology offloads full-lifecycle data management operations of virtual volumes to the storage system and uses the VASA API out-of-band management channel to implement vVol-based management. The following functions are provided:

Table 4-6 vVol-based management functions

vVol-based Management Function	Creating a VM
	Expanding the capacity of a VM disk
	Creating a snapshot for a VM
	Cloning a VM
	Fast cloning a VM
	Migrating a VM

The following figure shows the working principle of vVol snapshots and migration.

Figure 4-20 vVol of the OceanStor Pacific series structured service



1. Snapshot-related operations of virtual volumes can be performed on the storage system. The host does not need to pay attention to virtual volumes, simplifying management on the host.
2. If data needs to be migrated from VM 2 to VM 3, the data must pass through the host and network in traditional mode. By contrast, the vVol technology enables data to flow only on the storage system, occupying neither host CPU resources nor network bandwidth. In addition, local processing can reduce the data processing latency and improve the overall performance.

4.13 Scenario-specific SmartCompression (Scenario-specific Compression)

Based on unstructured semantics, the scenario-specific SmartCompression feature compresses and decompresses images in some specified formats and supports the file, object, and HDFS services. File formats and file metadata are identified based on file types in different scenarios and a proper compression algorithm is matched to compress a single file.

File semantics in unstructured data refer to a specific unstructured scenario to which a type of file belongs. Scenario-specific semantic compression automatically identifies file formats and file metadata in different scenarios for different types of files, performs lossless in-depth re-compression on the identified encoded data, and stores the data in a Huawei-dedicated format.

Scenario-specific SmartCompression supports only image formats in some scenarios, such as electron cryomicroscopy, genome sequencing, high-resolution

remote sensing, and medical pathology. The restrictions on image types are as follows:

Table 4-7 Restrictions on image types

Scenario	Supported File Types (Case-Insensitive)	Compression and Decompression
Electron cryomicroscopy	MRC and MRCS	By default, data is decompressed into uncompressed raw data and then re-compressed.
High-resolution remote sensing	TIFF and TIF	By default, data is decompressed into uncompressed raw data and then re-compressed.
Genome sequencing	FASTA and FASTQ FASTA.GZ and FASTQ.GZ	By default, data is decompressed into uncompressed raw data and then re-compressed.
Medical pathology	KFB, SVS, and SDPC	By default, data is decompressed into uncompressed raw data and then re-compressed.

Scenario-specific SmartCompression can be used together with the SmartTier feature. SmartTier allows low-value or seldom used files to be placed on low-cost devices with low performance and availability specifications (warm or cold storage). Scenario-specific SmartCompression allows files to be compressed and decompressed during data migration in tiered storage. The periodic scheduling logic of scenario-specific SmartCompression reuses the implementation of SmartTier. For details about terms such as disk pool and tiering, see the technical white paper of SmartTier.

4.14 Standard SmartCompression (Standard Compression)

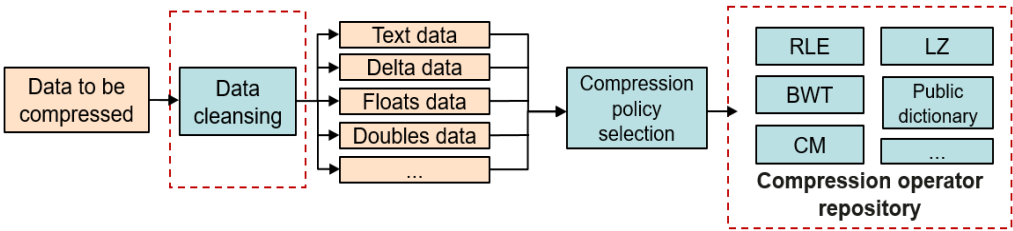
The standard SmartCompression feature is used for unstructured data. It compresses and decompresses user data and supports file, object, and HDFS services. The system automatically selects the optimal compression policy for file data in different scenarios. Except for videos, audios, images, encrypted data, compressed data, PDF files, and XML files, a sound compression ratio can be provided. [Table 4-8](#) lists typical application scenarios.

Table 4-8 Typical application scenarios of standard SmartCompression

Category	Scenario	Sub-scenario	Typical Data Format
HPC	Autonomous driving	Point cloud product	.bin\pcap\.
	Energy exploration	SEG-Y seismic data	.sgy
		Oil well simulation	.res\INC
	Meteorology & oceanography	Meteorology	.cdf\nc\hdf
	Manufacturing	CAD_CAE	.stl\nc\raw\off\op2
		EDA	.vpd\fsdb\sdb
	Education & scientific research	Computational chemistry and high energy physics	.csv\dc\.tif\.tif\txrm
		Brain science	.tif
		Astronomical data	.csv\fit\txt
Big data	parquet	Uncompressed parquet file	.parquet
Resource pool	Animation rendering	Animation rendering and post-production	.swf\scd
	Medical PCAS	db	.idb\dbf
		DCOM (non-compression format)	.dcm

The standard SmartCompression algorithm evolves from single-model to multi-model. Technologies such as data cleansing, compression policy selection, and operator repository are used to improve data reduction benefits, as shown in [Figure 4-21](#).

Figure 4-21 Multi-model standard SmartCompression algorithm



Standard SmartCompression provides enterprise customers with standard SmartCompression for both online services and tiering migration. These two work together to achieve the target compression ratio.

(1) Configure a standard SmartCompression policy for online services at the file system granularity. After standard SmartCompression is enabled for online services and when a user's service data is received, the data is divided at a certain granularity, compressed using a compression algorithm, and written into storage space to balance the compression ratio and efficiency, ensuring that space is immediately saved.

(2) Configure a standard SmartCompression policy for tiering migration at the file system granularity. After standard SmartCompression is enabled for tiering migration and when a tiering migration policy is configured and tiering migration is performed: if the raw data is compressed, the data generated upon compression is migrated; if the raw data is not compressed, the standard SmartCompression algorithm needs to be invoked to compress the data and the data generated upon compression is migrated to the target pool.

4.15 SmartMigration (Intelligent Data Migration)

4.15.1 File Migration Service

Intelligent file migration is mainly used to migrate data from heterogeneous storage devices. Currently, NAS data migration at the host layer is the most common migration solution. However, the following problems may occur:

- A Windows or Linux host is required as the migration server.
- The cutover time is long.

To solve the preceding problems, two intelligent file migration modes are provided:

- Copy-first

This mode is similar to host migration. Specifically, the built-in migration module of a storage system reads remote data and synchronizes the data to the local storage system. However, this mode does not require the customer to provide a Windows or Linux host as the migration server. In this mode, the built-in migration module and a NAS protocol client directly access the remote storage data in parallel. Similar to host migration, the cutover time in this mode depends on the time of the last incremental synchronization.

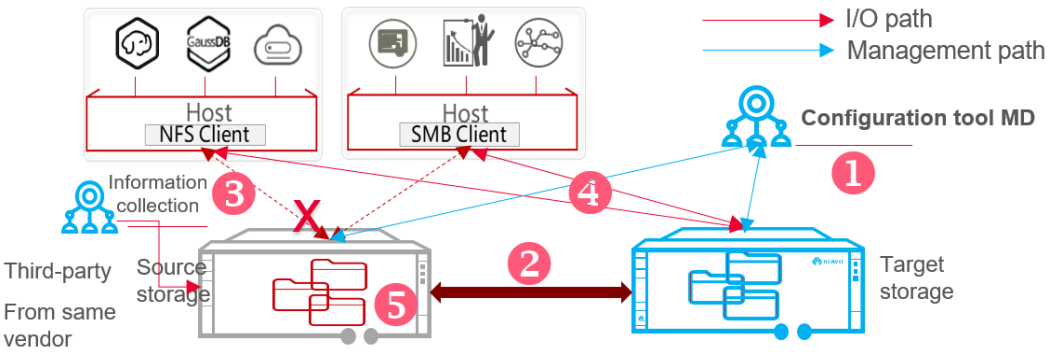
- Takeover-first

Different from traditional host migration, this mode shortens the cutover time to several minutes. In this mode, a customer stops services on the remote storage, creates a synchronization task between the local and remote storage, and switches I/Os to the local storage. In this way, the customer's services can continue and the generated I/O data is written to the local storage and also synchronized to the remote storage, ensuring data consistency. After a background data synchronization task migrates all data from the remote storage to the local storage, the synchronization task can be deleted. In this way, services can be seamlessly switched to the local storage. In this mode,

services can be migrated to the local storage using cutover without waiting for the completion of all data migration.

The following figure illustrates the copy-first mode.

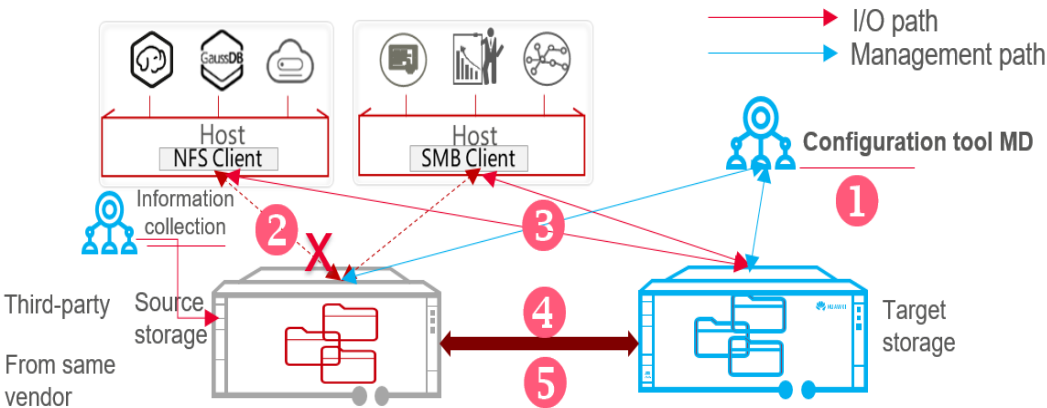
Figure 4-22 Process of data migration in copy-first mode



1. Preparation	Collect configuration information about the source storage and use MD to configure the target storage. Set up a connection between the source storage and the target storage and create a file system to be migrated.
2. Initial migration	Start initial migration using MD. The source storage continues to process client services.
3. Shutdown for cutover	Shut down the service client and start incremental migration based on difference comparison. After the migration, disconnect the two storage systems.
4. Service recovery	Complete IP address configuration and DNS modification, mount the shared file system to the service client, and use the target storage for read and write services.
5. Migration completion	Remove the source storage from the network.

The following figure illustrates the takeover-first mode.

Figure 4-23 Process of data migration in takeover-first mode



1. Preparation	Collect configuration information about the source storage and use MD to configure the target storage. Set up a connection between the source storage and the target storage and create a file system to be migrated.
2. Shutdown for cutover	Shut down the service client.
3. Service recovery	Complete IP address configuration and DNS modification, mount the shared file system to the service client, and write new services in dual-write mode (to both the target and source storage).
4. Service migration	Use MD to migrate data on the file system of the source storage in the background.
5. Service completion	Disconnect the two storage systems and remove the source storage from the network.

4.15.2 Volume Migration Service

With the advent and popularization of cloud, Internet, and intelligent technologies, mass data is generated continuously. Storage data migration faces severe challenges in scenarios such as data center consolidation and out-of-warranty storage. Take FusionStorage 6.3 as an example. FusionStorage 6.3 is widely used in resource pools and databases. After services are not provided for FusionStorage 6.3 anymore, OceanStor Pacific needs to be used as the replacement. How to optimize storage resources and implement data migration between storage systems becomes an urgent problem to be resolved.

By using the volume takeover and migration technology provided by SmartMigration, OceanStor Pacific allows you to migrate data from compatible heterogeneous storage systems to OceanStor Pacific without interrupting ongoing services. Currently, the supported heterogeneous storage system is FusionStorage 6.3.

The volume takeover and migration technology of SmartMigration is classified into takeover and online migration.

- The takeover technology takes over volume data access from heterogeneous storage systems to OceanStor Pacific. Takeover is classified into online takeover and offline takeover.
- Online migration of heterogeneous volumes completely migrates data on external volumes of heterogeneous storage systems to OceanStor Pacific.

4.15.2.1 Online Takeover

The online takeover technology takes over volume data access from heterogeneous storage systems to OceanStor Pacific. The entire service process is not interrupted, ensuring service continuity.

The local storage system accesses a heterogeneous storage system through the iSCSI protocol. If an application server accesses a storage system through the VBS client of the storage system, you need to switch the VBS client from FusionStorage 6.3 to that of OceanStor Pacific during the takeover.

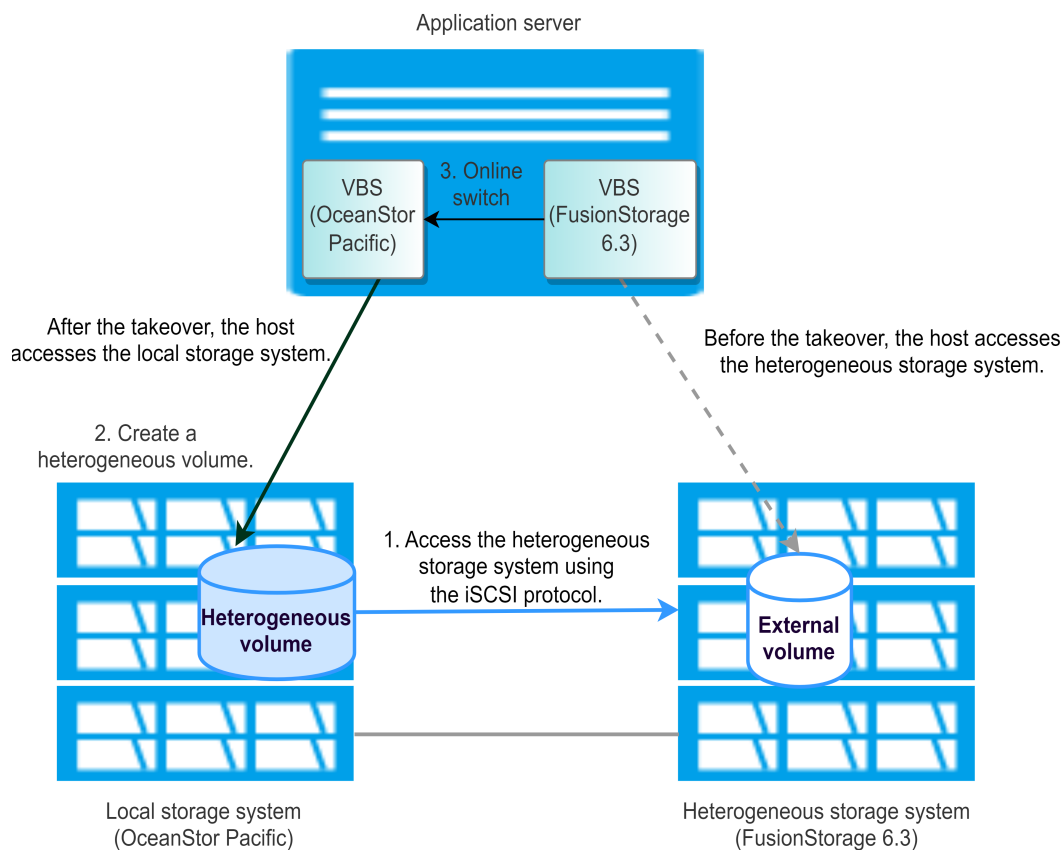
The online takeover process is as follows:

1. The heterogeneous storage system provides the iSCSI service, and the local storage system accesses the heterogeneous storage system using the iSCSI protocol.
2. Create a heterogeneous volume on the local storage system.
3. Switch the VBS block client of the application server online.

Before the takeover, host applications directly access the heterogeneous storage system. After the takeover, that is, after the VBS block client is switched, services of the host applications are accessed on the local storage system.

A heterogeneous volume on the local storage system matches an external volume on the heterogeneous storage system. Application servers can access data on external volumes by reading and writing heterogeneous volumes.

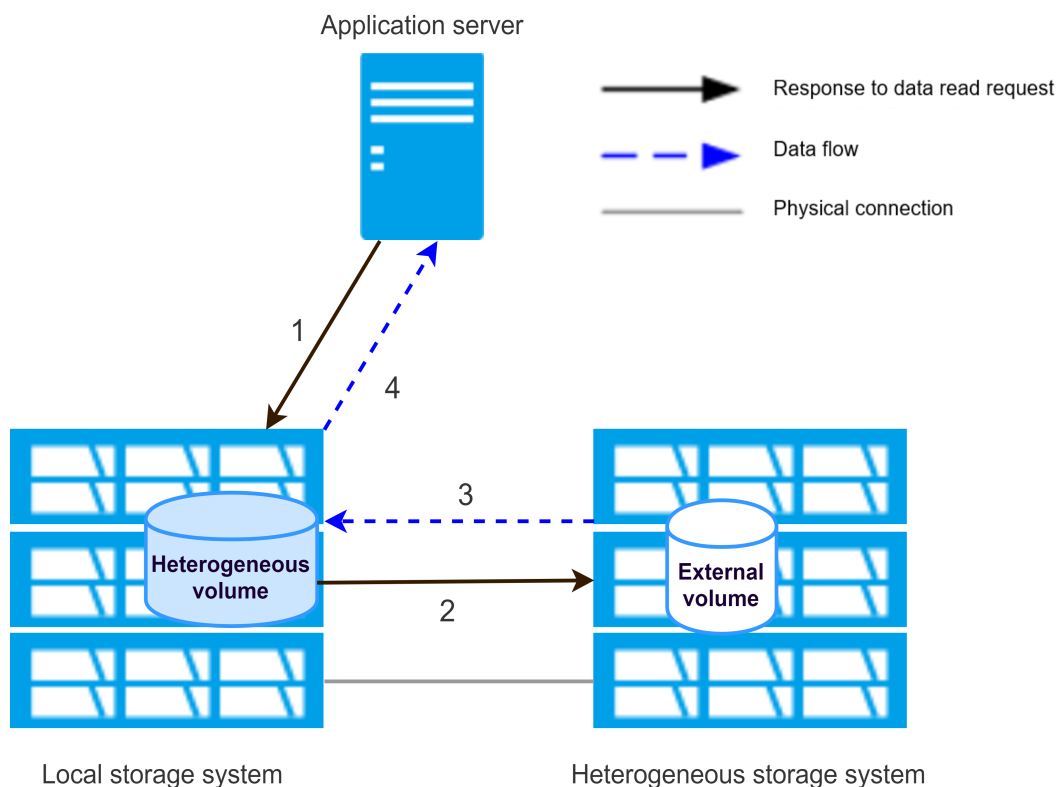
Figure 4-24 Online takeover



During the takeover, when a host accesses a heterogeneous volume of the local storage system, the host needs to access the heterogeneous storage system.

- Data read process:

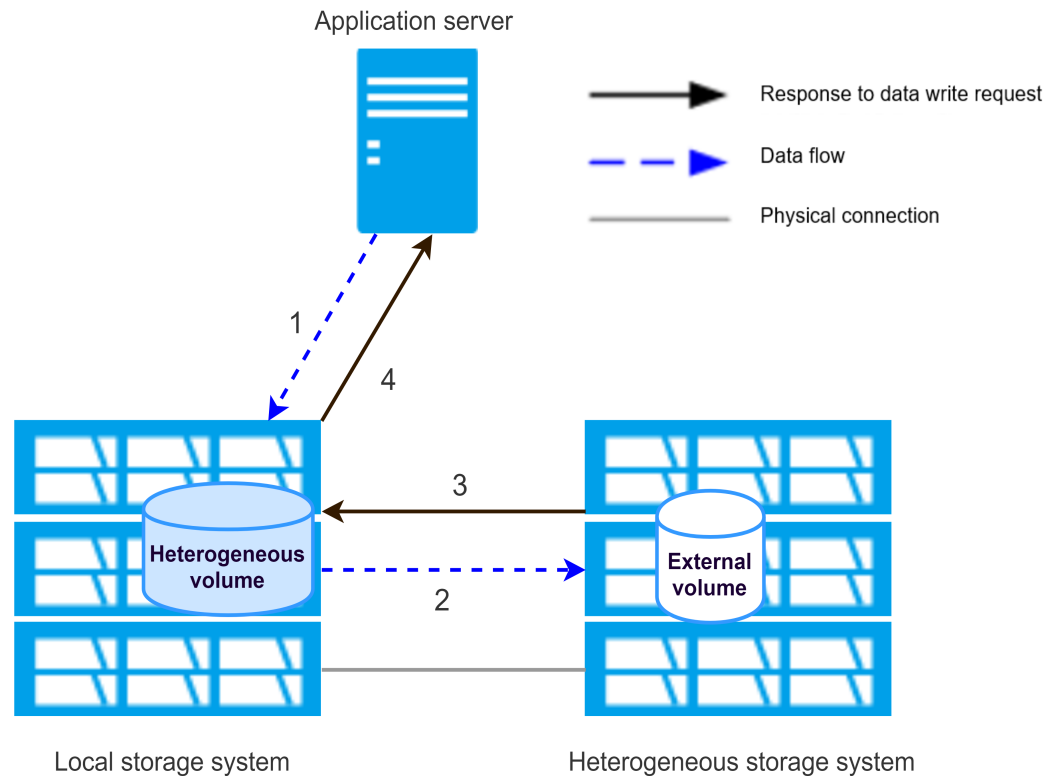
Figure 4-25 Data read process during the takeover



1. An application server initiates a data read request.
2. The local storage system forwards the data access request to the heterogeneous storage system.
3. The heterogeneous storage system returns the read result to the local storage system.
4. The local storage system returns the read data to the application server.

- Data write process:

Figure 4-26 Data write process during the takeover



1. An application server initiates a data write request.
2. The local storage system forwards the data write request to the heterogeneous storage system.
3. After data is written, the heterogeneous storage system returns a write response to the local storage system.
4. The local storage system returns a write success response to the application server.

4.15.2.2 Offline Takeover

For services involving interruption time windows, the offline takeover technology can be used to take over volume data access from heterogeneous storage systems to OceanStor Pacific. During offline takeover, services are interrupted for a certain period of time. After takeover is complete, services are recovered.

The main difference between offline takeover and online takeover is that during online takeover, VBS clients are switched online. During offline takeover, you need to delete the VBS client of FusionStorage 6.3 and then install that of OceanStor Pacific.

The offline takeover process is as follows:

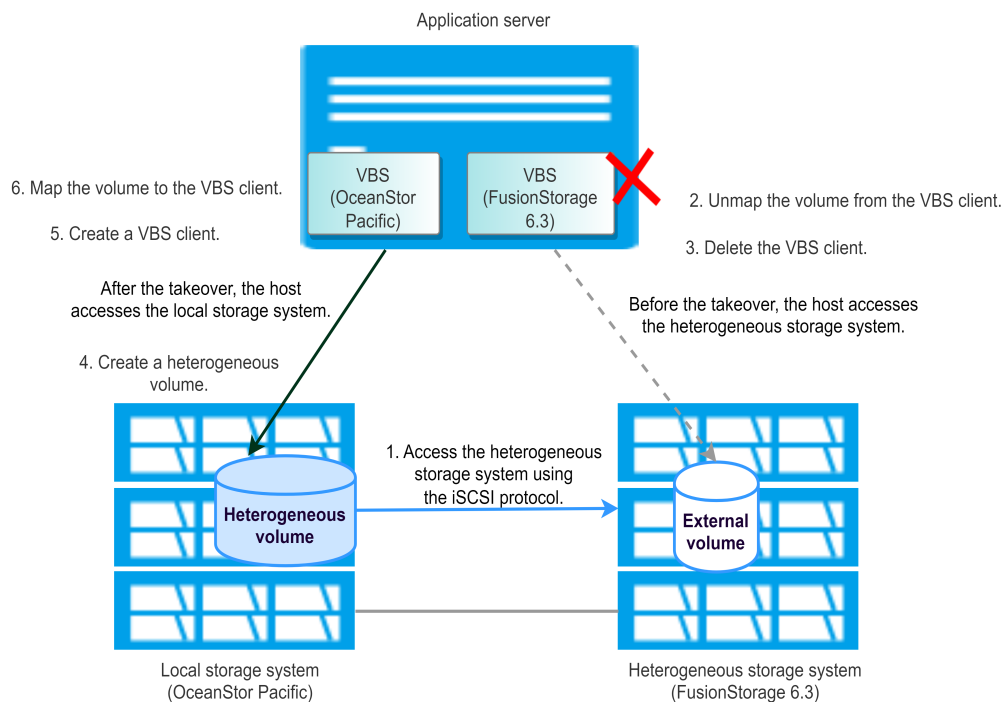
1. The heterogeneous storage system provides the iSCSI service, and the local storage system accesses the heterogeneous storage system using the iSCSI protocol.

2. Unmap the volume from the VBS client of FusionStorage 6.3 on the application server.
3. Delete the VBS client of FusionStorage 6.3 from the application server.
4. Create a heterogeneous volume on the local storage system.
5. Create a VBS client for OceanStor Pacific on the application server.
6. Map the volume to the VBS client of OceanStor Pacific.

Before the takeover, host applications directly access the heterogeneous storage system. After the takeover, services of the host applications are accessed on the local storage system.

During the takeover, when a host accesses a heterogeneous volume of the local storage system, the host needs to access the heterogeneous storage system. The data read/write process is the same as that of online takeover of heterogeneous storage systems.

Figure 4-27 Offline takeover



4.15.2.3 Online Migration of Heterogeneous Volumes

After external volumes of the heterogeneous storage system are taken over to the local storage system, data on the volumes of the heterogeneous storage system can be completely synchronized to OceanStor Pacific using the online volume migration technology.

The process of migrating heterogeneous volumes online is as follows:

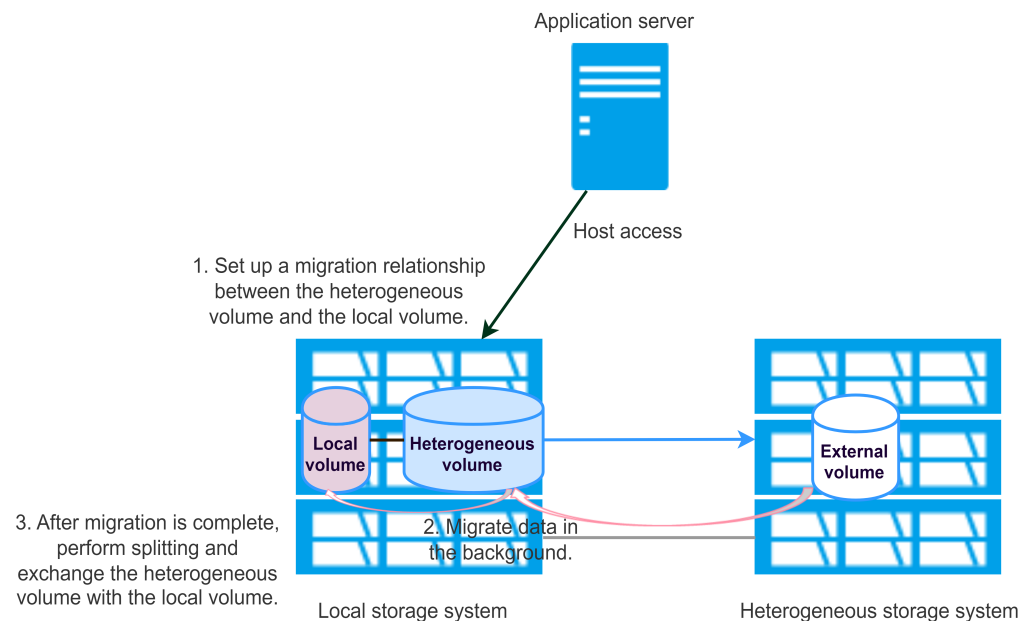
1. Create a local volume in the local storage system and set up an online migration relationship between the heterogeneous volume and the local volume.

The heterogeneous volume is the source volume, and the local volume is the target volume.

2. Synchronize data between the source volume and target volume in the background. That is, migrate data from the external volume of the heterogeneous storage system to the local storage system.

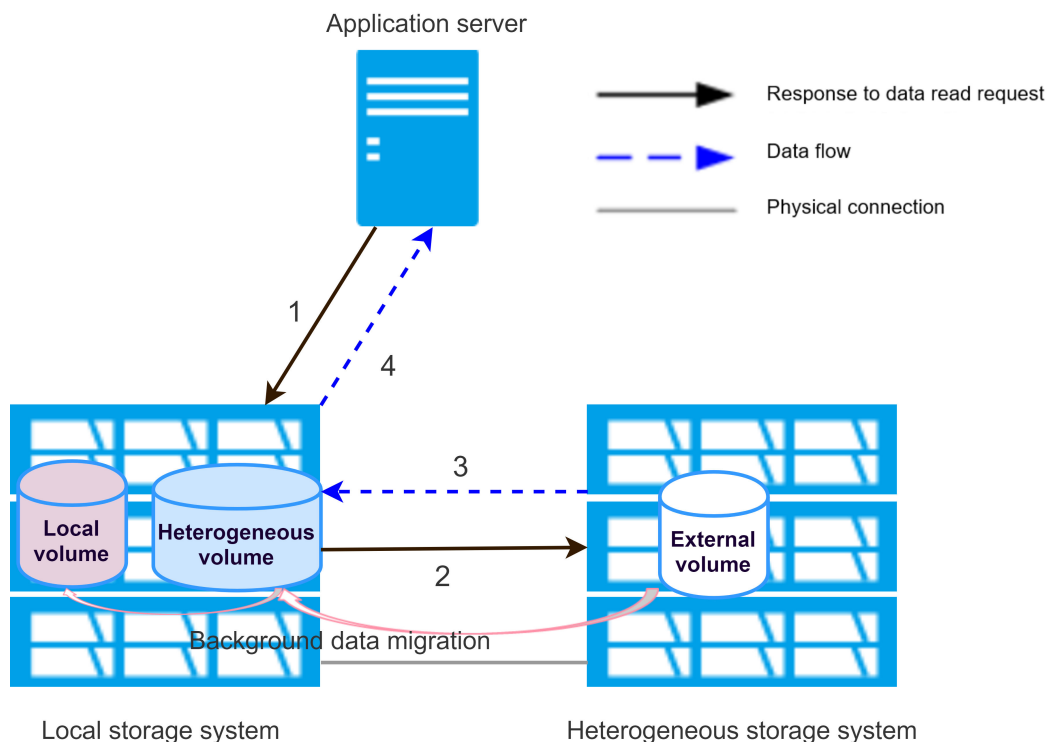
3. After data migration is complete, perform splitting. During splitting, exchange the heterogeneous volume with the local volume. After the exchange, hosts access only the local volume and do not access the heterogeneous storage system.

Figure 4-28 Online migration of a heterogeneous volume



Data read process: The data read process during the migration is the same as that during the takeover. The only difference is that background data migration also exists.

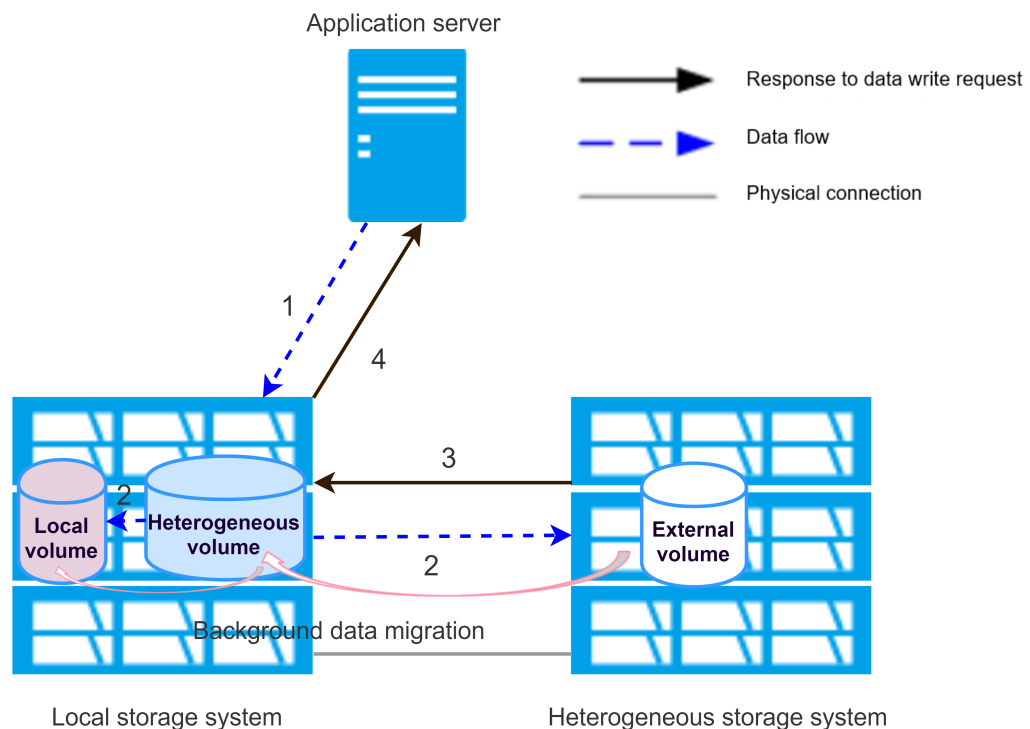
Figure 4-29 Data read process during the migration



1. An application server initiates a data read request.
2. The local storage system forwards the data access request to the heterogeneous storage system.
3. The heterogeneous storage system returns the read result to the local storage system.
4. The local storage system returns the read data to the application server.

Data write process:

Figure 4-30 Data write process during the migration



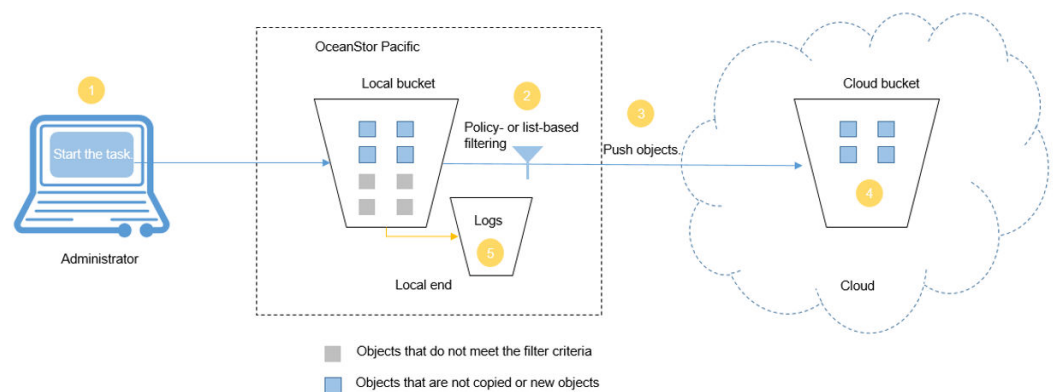
1. An application server initiates a data write request.
2. The local storage system receives the data write request and data is written to the heterogeneous storage system and the local volume in parallel.
3. After data is successfully written, the heterogeneous storage system returns a write response to the local storage system.
4. After data is successfully written to both the heterogeneous storage system and the local storage system, the local storage system returns a write success response to the application server.

4.16 SmartSync (Copying Objects to the Cloud)

As data becomes more and more important, customers have increasingly strong requirements for data flow. SmartSync adopts a built-in data transmission engine to transmit customer data between a local storage cluster and a remote storage device (such as OceanStor Pacific and cloud) through the S3 object protocol, ensuring data consistency between local and remote devices.

SmartSync supports the following data flow modes:

- **Data push**
Pushes object data from the local OceanStor Pacific system to buckets in the cloud (heterogeneous device) using the object protocol.



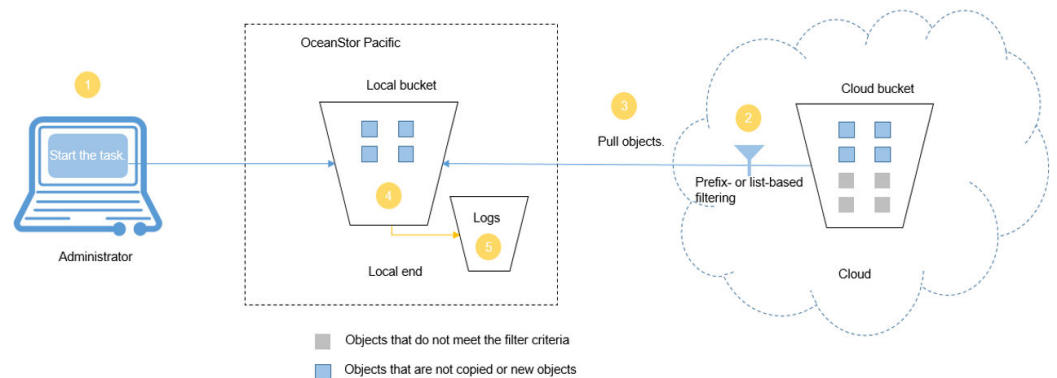
The data push mode supports the following data synchronization methods:

- One-off replication task: Object data in a bucket is replicated to the cloud at a time based on snapshots. In addition, specific object data can be replicated based on a specified object name list.
- Periodic incremental replication task: A periodic replication task is executed repeatedly based on the specified period. Only the data changed during the period is replicated each time. The replication period unit can be minute, hour, day, or month. The minimum period is 15 minutes and the maximum period is 12 months.

In data push mode, you can configure filter criteria for replication tasks, such as the object name, object size, operation time (supported only by one-off replication tasks), object name prefix, and tag.

- Data pull

Pulls object data from buckets in the cloud (heterogeneous device) to the local OceanStor Pacific system using the object protocol.



The data pull mode supports the following data synchronization method:

- One-off replication task: Object data in a bucket is pulled from the cloud to the local OceanStor Pacific system at a time.

For one-off replication tasks in data pull mode, specific object data can be replicated based on a specified object name list. You can also configure the object name prefix as the object filter criteria to synchronize data.

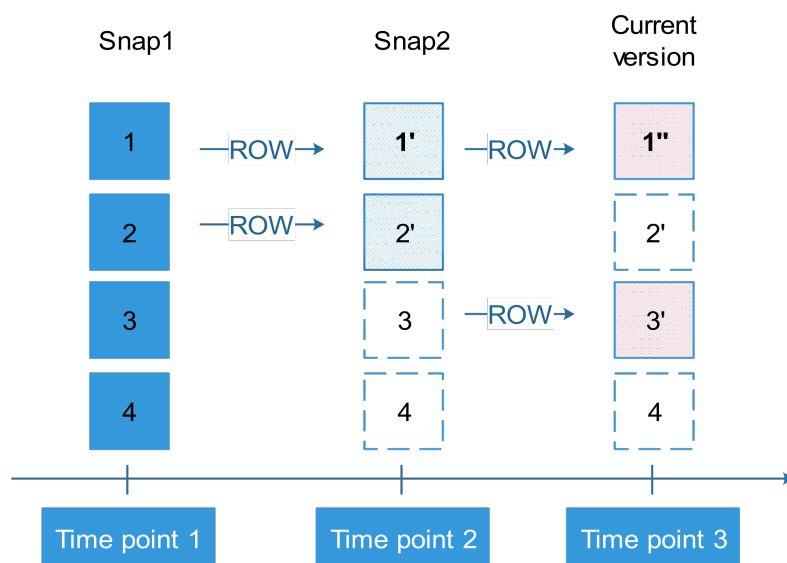
5 Value-added Feature: Data Protection

- 5.1 HyperSnap (Snapshot)
- 5.2 HyperReplication (Remote Replication)
- 5.3 HyperMetro (Active-Active)
- 5.4 HyperGeoMetro (Cross-Site Multi-Active of the Object Service)
- 5.5 HyperGeoEC (Cross-Site EC of the Object Service)
- 5.6 HyperClone (Linked Clone)
- 5.7 Recycle Bin
- 5.8 Antivirus
- 5.9 HyperLock (WORM)
- 5.10 NDMP Backup

5.1 HyperSnap (Snapshot)

OceanStor Pacific series supports second-level snapshots. Snapshot data is stored using the Redirect-On-Write (ROW) mechanism, and snapshots do not deteriorate the performance of the original volume.

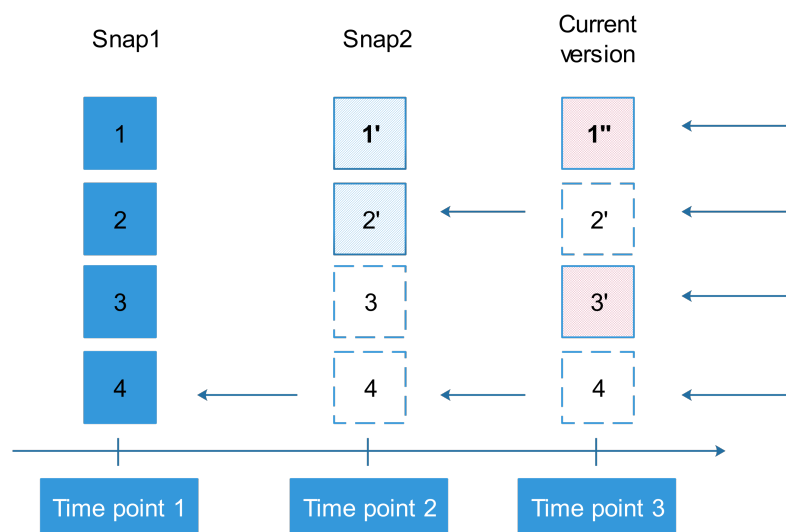
Figure 5-1 Working principle of ROW snapshots



As shown in the preceding figure, the ROW-based protection principle is as follows:

- After snapshot **Snap1** is created in a namespace at time point 1, new data is written to block 1 and block 2 in the namespace. The new data is written to new space (block 1' and block 2').
- No data is written to blocks 3 and 4 at time point 2 and therefore no new space is occupied.
- If snapshot **Snap2** is created at time point 2 and new data is written to block 1, the new data is written to block 1'' based on the ROW mechanism.
- When new data is written to block 3, it will be written to block 3' and data in snapshots **Snap1** and **Snap2** remains unchanged.

Figure 5-2 Read-protected process of snapshot data



As shown in the preceding figure, when you access file system data after snapshot **Snap2** is created:

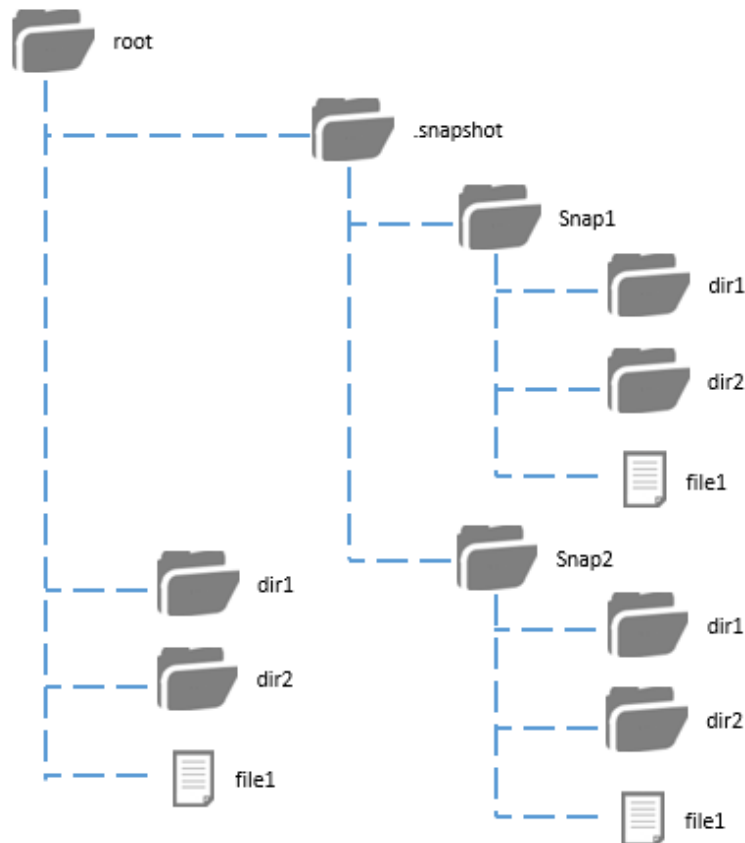
- For block 1, you directly access block 1'' because new data is generated in the current version.
- For block 2, you access block 2' in snapshot **Snap2** because no new data is generated in the current version.
- For block 3, you directly access block 3' because new data is generated in the current version.
- For block 4, you access block 4 in snapshot **Snap1** because block 4 remains unchanged since snapshot **Snap1** is created.

5.1.1 Unstructured Data Storage Services

Unstructured services of OceanStor Pacific series allow you to quickly obtain a copy that is consistent with the source namespace without interrupting the normal services of the source namespace. Creating a snapshot for a namespace is equivalent to creating a protected read-only version for the namespace. When you write data to or delete data from a file system, the existing data space is protected by the snapshot and thus remains unchanged. New data will be written

to new storage space in ROW mode. The OceanStor Pacific series file service supports namespace-level and dtree-level snapshots. The two types of snapshots can be nested (only two layers are allowed). Snapshots of a namespace can protect all dtrees in the namespace. You can create up to 4096 user snapshots and 4864 scheduled snapshots for each namespace or dtree.

Figure 5-3 Snapshot directory



After a snapshot is created, sub-directory **.snapshot** is generated in the snapshot root directory. A snapshot sub-directory (the sub-directory name is the snapshot name) is generated for each snapshot in the **.snapshot** directory. Each subdirectory is the access entry of the snapshot version. For example, after snapshots **Snap1** and **Snap2** are created for the namespace, the **.snapshot** directory and its snapshot subdirectories are generated in the root directory of the namespace, as shown in the preceding figure. Accessing the directories or files in the **.snapshot** directory is to access the directories or files of the snapshot version. When accessing the snapshot data, users access the data at a specific time point according to the time point corresponding to the snapshot ID.

5.1.2 Structured Data Storage Service

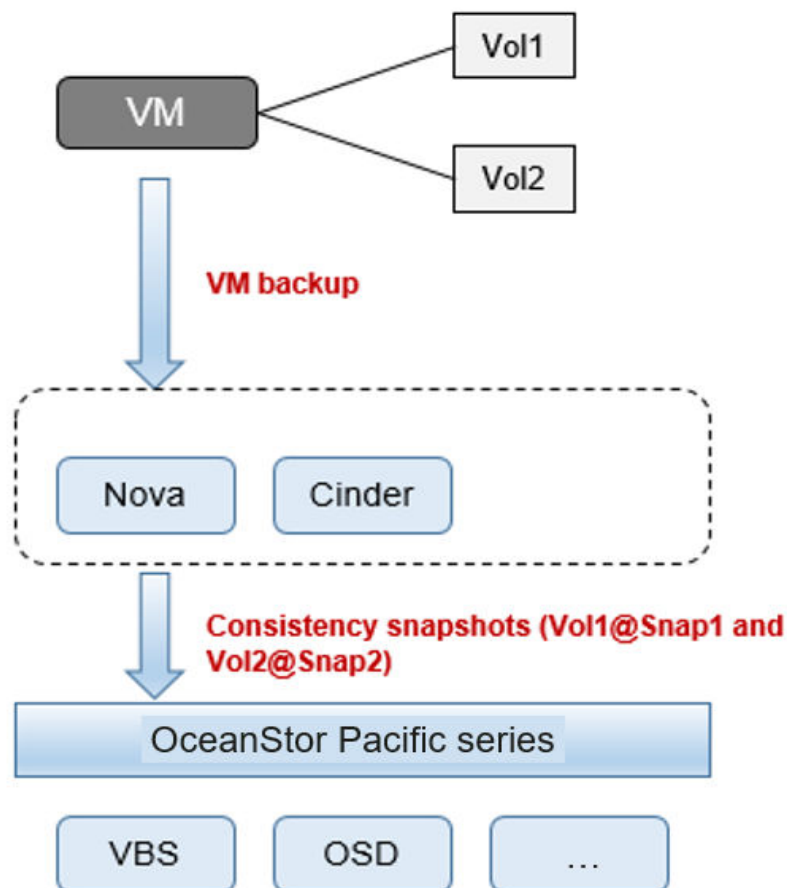
The structured service of OceanStor Pacific series provides a second-level snapshot mechanism to save the status of volume data at a certain point in time. The data can be exported and used for data recovery.

To back up shared volumes, the structured service of OceanStor Pacific series also supports snapshots for shared volumes. A SCSI volume with multiple mount points

is called a shared volume. However, all iSCSI volumes are shared volumes. The procedure for creating a snapshot for a shared volume is the same as that for creating a snapshot for a common volume.

Consistency snapshots are used for VM backup. A VM is usually mounted with multiple volumes. When a VM is backed up, all volume snapshots must be at the same point in time to ensure the reliability of data recovery. The structured service of OceanStor Pacific series supports the consistency snapshot capability. The consistency snapshot requests initiated by the upper layer ensure that the snapshots of multiple volumes are created at the same time point. The structured service of OceanStor Pacific series invalidates multiple volumes at the current point in time and then updates the snapshot information to ensure that the snapshots of multiple volumes are at the same point in time.

Figure 5-4 Structured consistency snapshot of OceanStor Pacific series



5.2 HyperReplication (Remote Replication)

As digitization advances in various industries, data has become critical to the efficient operation of enterprises, and users impose increasingly demanding requirements on the stability of storage systems. Although many enterprises have highly stable storage systems, it is a big challenge for them to ensure data recovery from damage caused by natural disasters. To ensure continuity, recoverability, and high availability of service data, remote DR solutions emerge.

The remote replication technology is one of the key technologies used in remote DR solutions.

Unstructured services of OceanStor Pacific series provide namespace- and dtree-based asynchronous replication and the structured service provides volume- and consistency group-based asynchronous replication and synchronous replication.

For asynchronous replication, incremental synchronization based on read-only snapshots is periodically performed for data on primary and secondary storage systems to implement system DR, minimizing service performance deterioration caused by the latency of remote data transmission.

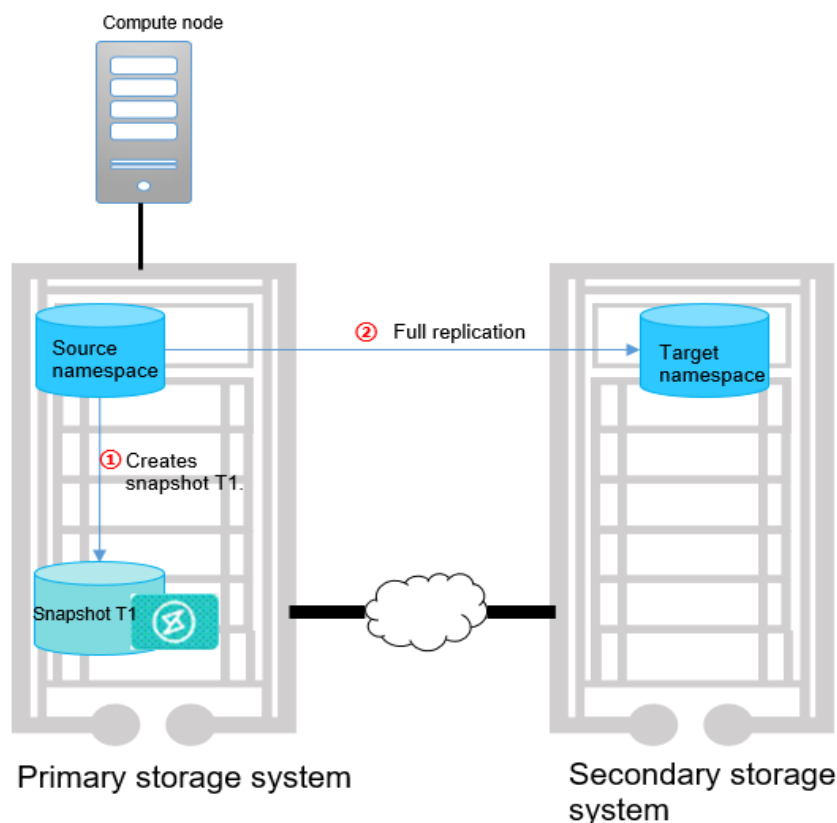
For synchronous replication, a write success message is returned to a host only after data is successfully written to both the production storage and DR storage, ensuring that the RPO is 0. Synchronous replication implements data-level DR with a high DR level (level 6: zero data loss and remote cluster support).

NOTE

A consistency group is a collection of multiple remote replication pairs. It ensures data consistency when a host writes data to multiple volumes in a single storage system. In medium- and large-sized database applications, data, logs, and modification information are stored on different volumes in the storage system. If data on one of the volumes is unavailable, data on the other volumes is also invalid. Based on this scenario, the structured service of OceanStor Pacific series provides asynchronous replication based on consistency groups to ensure data consistency between multiple remote replication volumes.

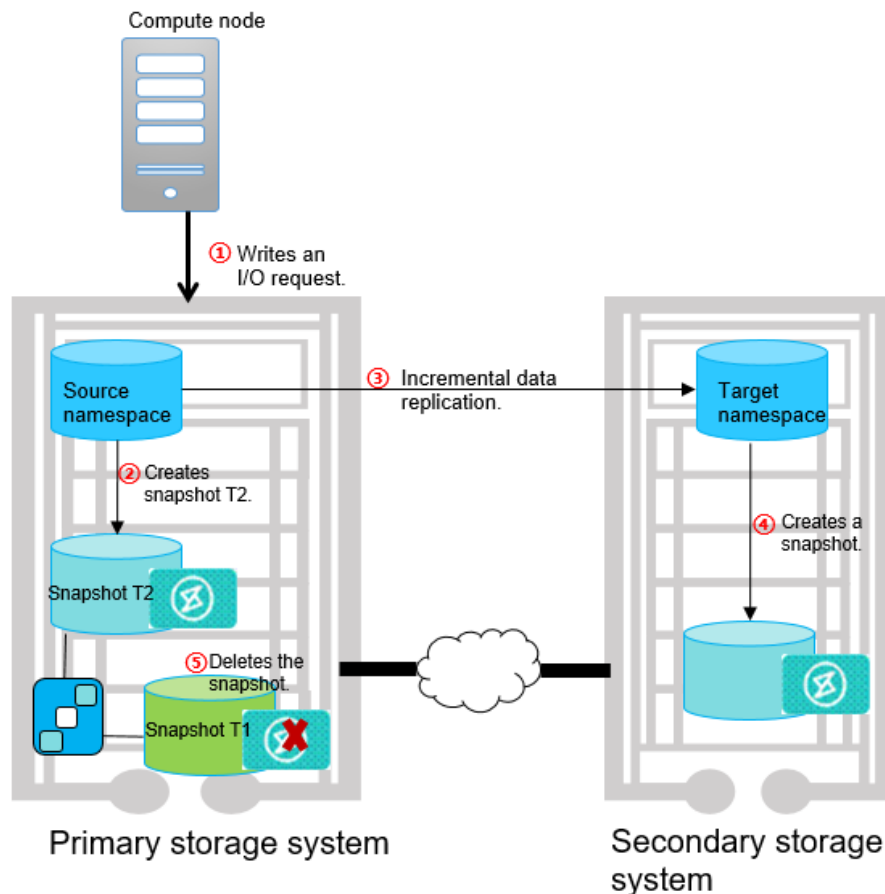
Asynchronous replication is divided into two phases. The first phase is the initial synchronization when the replication relationship is created. After the initial synchronization is complete, the second phase is the periodic incremental synchronization. The following uses asynchronous replication of namespaces as an example to describe the working principle:

Figure 5-5 Initial full synchronization of asynchronous replication



After a remote replication relationship is set up between source and target namespaces, initial synchronization is started. Before the initial synchronization, snapshot **Snapshot T1** is created for the source namespace. All data on the snapshot at the point in time is copied to the target namespace to ensure data consistency between target and source namespaces (a snapshot created at a point in time).

Figure 5-6 Incremental synchronization of asynchronous replication



After the initial synchronization is complete, incremental data is synchronized based on differential snapshot logs, as shown in the preceding figure.

1. The host sends a write I/O request to the primary namespace.
2. Create snapshot **Snapshot T2** before starting asynchronous replication.
3. The differential data between snapshots **Snapshot T2** and **Snapshot T1** is synchronized to the secondary storage system.
4. After data synchronization is complete, create a snapshot for the target namespace on the secondary storage system.
5. Delete the snapshot created when the last synchronization is complete from the secondary storage system. The previous snapshot is also deleted from the primary storage system.

Synchronous replication (supported only by the structured service) is divided into two phases:

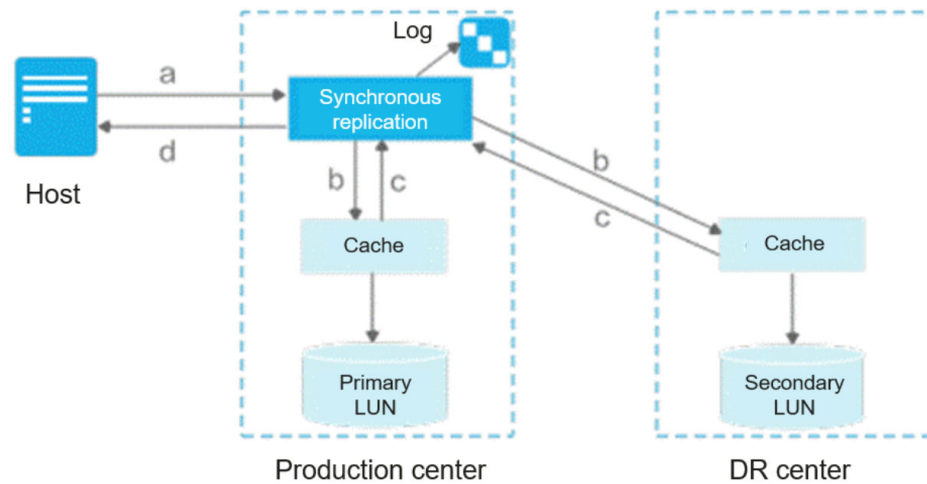
Phase 1: Initial synchronization is performed when the replication relationship is set up.

Phase 2: After the initial synchronization is complete, the primary and secondary LUNs enter the dual-write state. Each host write I/O is concurrently written to both the primary and secondary LUNs. A write success message is returned to the host only after the data is successfully written to the primary and secondary LUNs. Therefore, synchronous remote replication can achieve zero RPO.

The working principle is as follows:

1. Initial synchronization: After a synchronous remote replication relationship is set up between the primary and secondary LUNs, initial synchronization is started.
 - All data on the primary LUN is copied to the secondary LUN.
 - During initial synchronization, if the primary LUN receives a write request from a host and data is written to the primary volume, the data is also written to the secondary LUN.
2. Dual-write state: The primary and secondary LUNs enter the normal state after the initial synchronization is complete. In this state, data on the primary LUN is the same as that on the secondary LUN. The I/O process in the normal state is as follows:

Figure 5-7 Synchronous remote replication

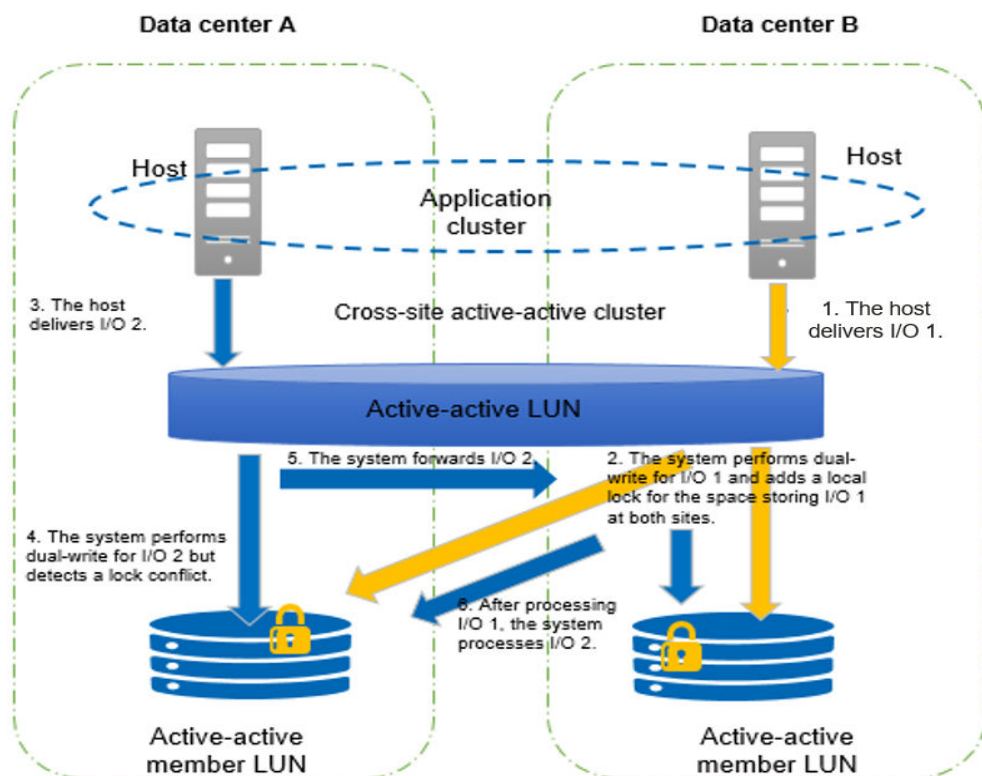


3. The production storage receives a write request from a host. Synchronous remote replication logs the request. The log only records the address information but no data content.
4. The data of the write request is written to both the primary and secondary LUNs. Generally, data is written to the cache.
5. Synchronous remote replication waits for write results from both the primary and secondary LUNs. If writing data to the secondary LUN times out or fails, the synchronous remote replication relationship is interrupted. If the data is successfully written to both primary and secondary LUNs, the log is cleared. Otherwise, the log is retained and the request data is written into the data change log (DCL). In this case, the remote replication pair enters the disconnected state. When the synchronization is started later, the data blocks corresponding to the log address are replicated again.
6. The processing result of the write request is returned to the host. The processing result is subject to that of the primary LUN. If the data fails to be written to the primary LUN, a failure message is returned to the host even if the data is successfully written to the secondary LUN.

5.3 HyperMetro (Active-Active)

The structured service of OceanStor Pacific series provides the HyperMetro feature to build an active-active DR relationship based on two OceanStor Pacific series storage clusters in data centers A and B. An active-active volume is virtualized based on the volumes of the two OceanStor Pacific series storage clusters. Hosts in the two data centers can read and write the volumes at the same time. If one data center fails, the other automatically takes over services without data loss and service interruption.

Figure 5-8 HyperMetro principle of the OceanStor Pacific series structured service



The structured service of OceanStor Pacific series provides HyperMetro for incremental synchronization. If a site fails, the site winning arbitration continues to provide services. I/O requests change from the dual-write state to the single-write state. After the faulty site recovers, incremental data can be synchronized to it to quickly restore the system.

The structured service of OceanStor Pacific series supports logical write error handling. If the system is running properly but one site fails to process a write I/O, the system will redirect the write I/O to a normal site for processing. After the fault is rectified, incremental data can be synchronized from the normal site to the one that fails to process the I/O. By doing so, upper-layer applications do not need to switch sites for I/O processing upon logical write errors.

The HyperMetro feature of the OceanStor Pacific series structured service can work with upper-layer applications such as Oracle RAC and VMware. It is recommended that the deployment distance in database scenarios be less than

100 km and that in VMware scenarios be less than 300 km. For details about the compatibility, visit Huawei Storage Interoperability Navigator at <https://info.support.huawei.com/storage/comp/#/home>.

5.4 HyperGeoMetro (Cross-Site Multi-Active of the Object Service)

Cross-site DR includes cross-site DR at the application layer and that at the data layer. Generally, cross-site DR at the application layer is provided by an application service provider, while cross-site DR at the data layer is provided by a storage system.

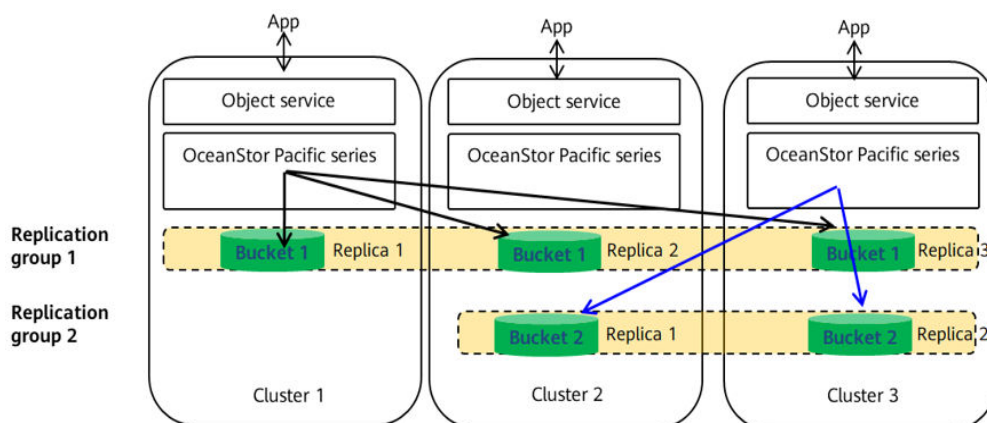
A typical cross-site DR system consists of a production center and a DR center. Factors such as the distance between the two centers, network, and service requirements directly affect the replication technology selection. Synchronous replication has strict requirements on the distance (usually less than or equal to 100 km), network latency (usually less than or equal to 2 ms RTT), and network bandwidth. So, usually, this technology is used only for transactions and other high-value services. It is of high cost and supports a limited number of sites (usually two sites or two centers), but its recovery point objective (RPO) is 0. On the contrary, asynchronous replication has lower requirements on the distance, network latency, and network bandwidth, but its RPO is greater than 0.

With the development of object service applications, some applications need to write and read data, from different sites, to and from the same bucket, so that the traditional single-active DR is unable to meet the application requirements. To deal with these requirements, the object service of OceanStor Pacific provides cross-site DR with multi-active objects, allowing different applications to write and read data, from different sites, to and from the same bucket.

Cross-site DR supports asynchronous replication of user data to a remote site. If a service site is faulty (such as overall power outage due to an earthquake or another natural disaster) or unable to provide services because of site maintenance, upper-layer applications can be automatically switched to the remote site to continue data reading and writing, ensuring service continuity.

HyperGeoMetro allows you to configure redundancy policies by bucket. As shown in the following figure, three replicas are configured for bucket 1 and two replicas are configured for bucket 2, thus providing more flexible choices for applications.

Figure 5-9 Cross-site multi-active at the bucket level



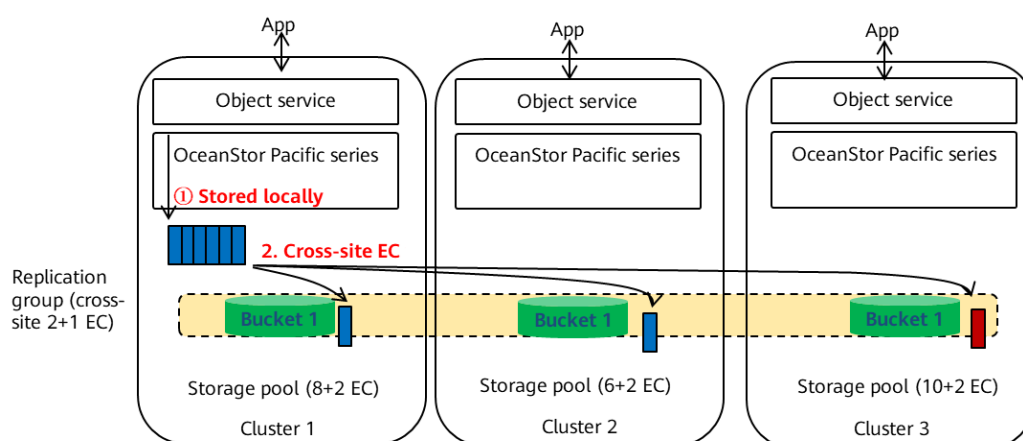
5.5 HyperGeoEC (Cross-Site EC of the Object Service)

The object service has high requirements on data reliability and costs in archiving scenarios. In addition to remote data DR, the cost needs to be reduced as much as possible. HyperGeoEC implements cross-site data storage in EC mode and supports multiple redundancy granularities. Compared with the replica mode, the EC mode reduces costs by more than 50%, and also meets reliability and cost requirements.

Similar to HyperGeoMetro, HyperGeoEC allows you to set the redundancy ratio by bucket. For different buckets, you can select different redundancy ratios, and choose the replica mode or EC mode.

In addition to inter-site redundancy, HyperGeoEC also provides intra-site redundancy. If a disk or node at a site is faulty, intra-site redundancy can be used. In this case, cross-site data rebuilding is not required, reducing the dependency on inter-site networks and improving reliability.

Figure 5-10 Cross-site EC

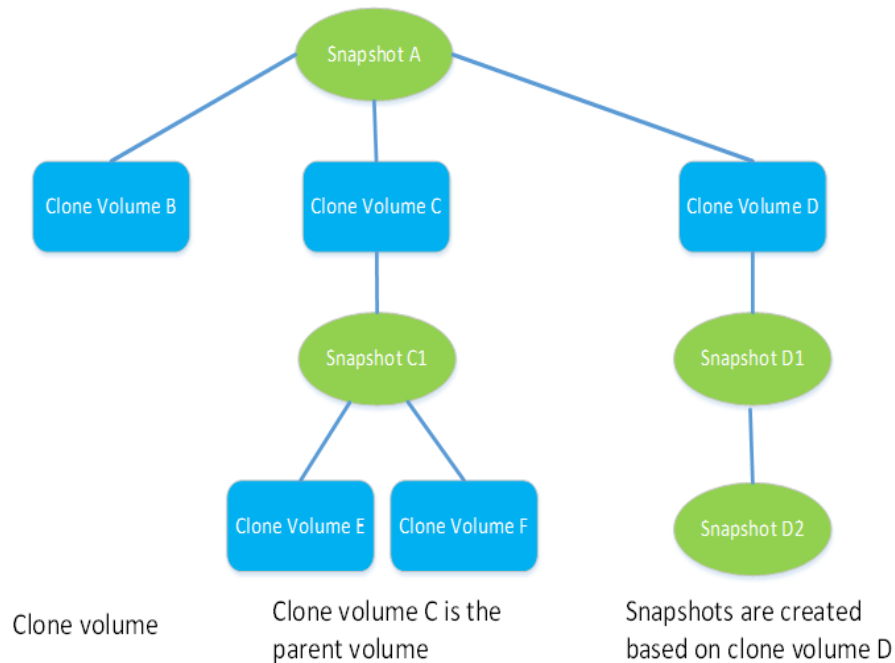


5.6 HyperClone (Linked Clone)

The structured service of OceanStor Pacific series provides the HyperClone feature based on the linked clone mechanism. Multiple clone volumes can be created

based on one volume snapshot. The data content of each clone volume is the same as that of the volume snapshot. Subsequent modifications to a clone volume do not affect the original snapshot and other clone volumes. A linked clone ratio of up to 1:2048 is supported, improving storage space utilization. A clone volume has all functions of a common volume. You can create snapshots for a clone volume, use the snapshots to restore the clone volume, and clone the clone volume.

Figure 5-11 Linked clone of OceanStor Pacific series



5.7 Recycle Bin

Unstructured services of OceanStor Pacific series provide the namespace/dtree-based recycle bin. When a user deletes files, the files are moved to the recycle bin directory in the background so that the user can retrieve the data. After the files are moved from their original locations to the recycle bin, they will not be actually deleted and storage space will not be released. To release space in a timely manner, administrators can configure a retention period for the files in the recycle bin. The system will automatically delete the files that have been retained for a period longer than the retention period from the recycle bin.

A recycle bin directory **.recyclebininternal** is created for each namespace or dtree. When a file is deleted, its data is not deleted. Instead, the data is moved to the recycle bin in the background, and the relative path before the file is deleted is retained in the recycle bin. For example, if the relative path of a file belonging to user **root** in a dtree is **/a/b/c.txt**, the path of the file in the recycle bin is **./recyclebininternal/root/current/a/b/c.txt**. The directories and files in the recycle bin that can be accessed by users are controlled by the recycle bin operation permission. Common users can access their own directories and files. The recycle bin is a special directory in a file system. User **root** can also grant permissions so that other users can access the recycle bin.

The system periodically checks the directories and data in the recycle bin. When a file is moved to the recycle bin, it is stored in the `/.recyclebininternal/User directory/current` directory first. The system renames the **current** directory to a directory named after the current time, and then checks the difference between the current time and the time when the directory is created to determine whether the retention period expires. If the retention period expires, the system will delete the directory.

NOTE

The recycle bin function is disabled by default. You can enable or disable it when creating a namespace or modify its status when using a namespace as required. The modification takes effect only for files that are deleted later.

5.8 Antivirus

As more and more user files are being stored in storage systems, these files must be protected against virus infection by using antivirus software.

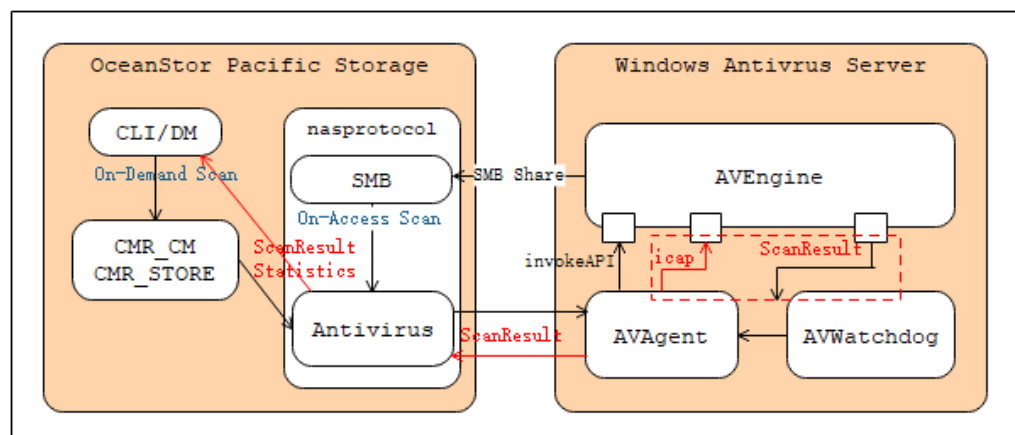
Antivirus is an important value-added feature supported by OceanStor Pacific series products used to protect NAS storage security. The antivirus feature uses third-party antivirus software for both on-access and on-demand file scanning. Users can configure scanning policies and scanning servers, and export and dump antivirus logs.

The system supports both on-demand and on-access virus scanning. They interact with the antivirus engine through API invoking or ICAP.

- On-demand virus scanning: The system scans for viruses based on the specified policy, period, and time window.
- On-access virus scanning: When a file is opened or closed, the system scans the file for viruses in real time.

The following is the processes of the two scanning methods.

Figure 5-12 Working principles of OceanStor Pacific series antivirus scanning



Interaction with the antivirus engine:

1. API invoking: The AvAgent program installed on the AV server invokes the API to instruct the antivirus software to scan viruses. The antivirus software uses

the SMB protocol to read necessary file data for scanning based on its logic for determining virus characteristics. This reduces the amount of data to be transmitted and improves the scanning speed.

2. ICAP: After receiving a file scanning request, the AvAgent uses ICAP to send the request to the antivirus software for virus scanning.

5.9 HyperLock (WORM)

Thanks to the rapid development in the information era, various types of data are increasing day by day. Protecting critical data from accidental deletion or modification is a key service requirement of most enterprises. For example, important data, such as court case files, medical records, and financial files, must be read-only within a specific period and cannot be modified. Protection measures must be taken on such files. When the files are protected, no one can delete or modify them. OceanStor Pacific provides the HyperLock feature, that is, write once read many (WORM), to implement this function, effectively ensuring the security of important data.

WORM Clock

A WORM clock is used to time WORM files. After a WORM clock is set, the system times protection periods according to the clock. This ensures that files are properly protected even if the local clock time is changed.

A WORM clock can only be set once, and cannot be modified after being set.

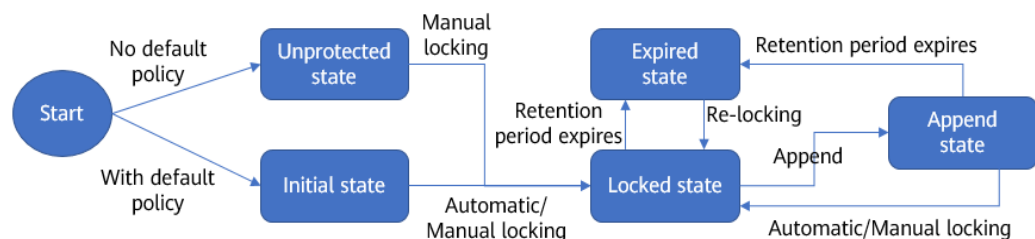
A WORM clock can automatically adjust its time according to the local node time:

- If the local node time is earlier than the WORM clock time, the WORM clock winds back its time 138 seconds or less every hour.
- If the local node time is later than the WORM clock time, the WORM clock adjusts its time to the local node time.

WORM File States

The following figure shows the transition between WORM file states.

Figure 5-13 WORM file states



- Unprotected: Files in the unprotected state can be modified or deleted, like common files.
- Initial: After an object is uploaded, it enters the locked state immediately. For a file, if the interval between the last modification time and the current time

exceeds the preset automatic submission delay, the file is automatically locked. The default retention period is used for automatic locking.

- Locked: Files in the locked state can be read, but cannot be deleted or modified.
- Expired: After the retention period of a file expires, the file enters the expired state. In this case, the file can be deleted and read, but cannot be modified.
- Append: After the write permission is added to an empty file in the locked state, this file enters the append state. Protected data that has been written to the file cannot be modified, but data can be appended to the file. After the write permission is disabled, objects are automatically locked after the append operation is complete, but files are automatically locked when the interval between the last modification time and the current time exceeds the preset automatic submission delay. If the append state does not end, the expired state starts when the retention period expires.

5.10 NDMP Backup

As digital information becomes the trend, users attach more importance to data and are more demanding on data security. Data loss due to hardware faults and misoperations must be prevented. Data backup is a means to retain data and use the retained data to recover data lost after a system damage or other faults. Data backup prevents data loss or corruption caused by unexpected accidents, ensuring data security.

Network Data Management Protocol (NDMP) is an open protocol used for enterprise-class data management. NDMP is used to control backup, recovery, and data transfer between primary and secondary storage. NDMP works based on the client/server model. OceanStor Pacific functions as the NDMP server and a data management application (DMA) functions as the client. OceanStor Pacific exchanges packets in compliance with NDMP V4.

Before backup, OceanStor Pacific generates a private snapshot for the file system to be backed up, and then performs file-level backup using the snapshot. The following backup modes are supported:

- Full backup: All required data is backed up to a tape library.
- Differential incremental backup: This backup mode is applicable to file-level backup. Only the files changed since the last backup are backed up.
- Cumulative incremental backup: This backup mode is applicable to file-level backup. Only the files changed since the last full backup are backed up.

OceanStor Pacific supports the following recovery modes:

- Full recovery: All data is recovered to its original locations.
- Direct access recovery (DAR): Backup software reads backup data from a tape library and then uses the backup data to recover the user-defined data.
- Non-DAR recovery: Backup software finds data to be recovered in a tape library and then recovers it.

The NDMP backup feature of OceanStor Pacific also supports the following functions:

- Multi-stream concurrency: A maximum of 16 backup or recovery tasks can be concurrently executed, improving backup and recovery efficiency.
- File and directory filtering: Backup can be performed based on the preset file and directory filtering conditions.

Supported networking modes:

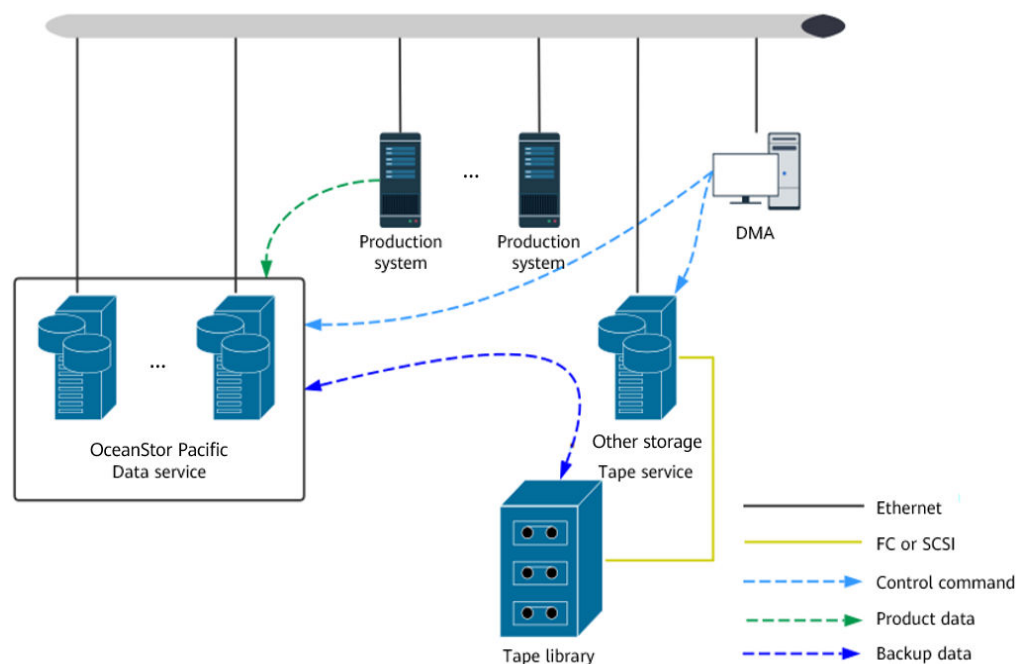
- 3-way networking

In 3-way backup networking, backup media are directly connected to a storage system (secondary storage) other than OceanStor Pacific through a SCSI or FC link. The storage system (secondary storage) that connects to the tape library provides the tape service and the OceanStor Pacific storage system (primary storage, which functions as the data backup source) provides the data service.

The DMA sends backup or recovery instructions to the corresponding primary storage (OceanStor Pacific) and secondary storage (other storage) over the network to start or stop backup or recovery.

- In backup scenarios, the primary storage sends data to the secondary storage and the secondary storage writes the data to the tape library.
- In recovery scenarios, the secondary storage reads data from the tape library and sends the data to the primary storage.

Figure 5-14 3-way networking of NDMP



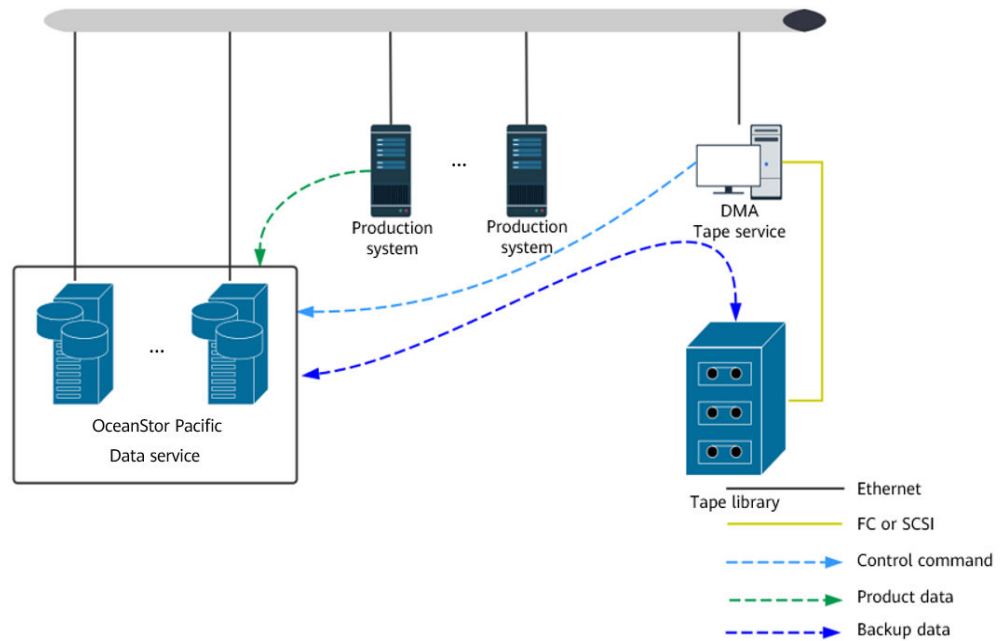
- Remote networking

In remote backup networking, backup media are directly connected to the server where the DMA resides. The DMA is responsible for operations related to the tape library.

- In backup scenarios, the primary storage sends data to the DMA and the DMA writes the data to the tape library.

- In recovery scenarios, the DMA reads data from the tape library and sends the data to the primary storage.

Figure 5-15 Remote networking of NDMP



6 End-to-End Performance Optimization

Huawei OceanStor Pacific series storage system uses the access balancing, I/O load balancing, and high-speed network designs for compute nodes, and use FlashLink technology to optimize hybrid I/O loads and end-to-end system performance based on full I/O paths, providing customers with ultra-high IOPS and low latency. A single OceanStor Pacific 9950/9920 device provides up to 160 GB/s bandwidth and 6.4 million IOPS for the file service. A single OceanStor Pacific 9920 node provides up to 210,000 IOPS for the block service at a stable latency of 1 ms.

Figure 6-1 OceanStor Pacific series end-to-end performance optimization

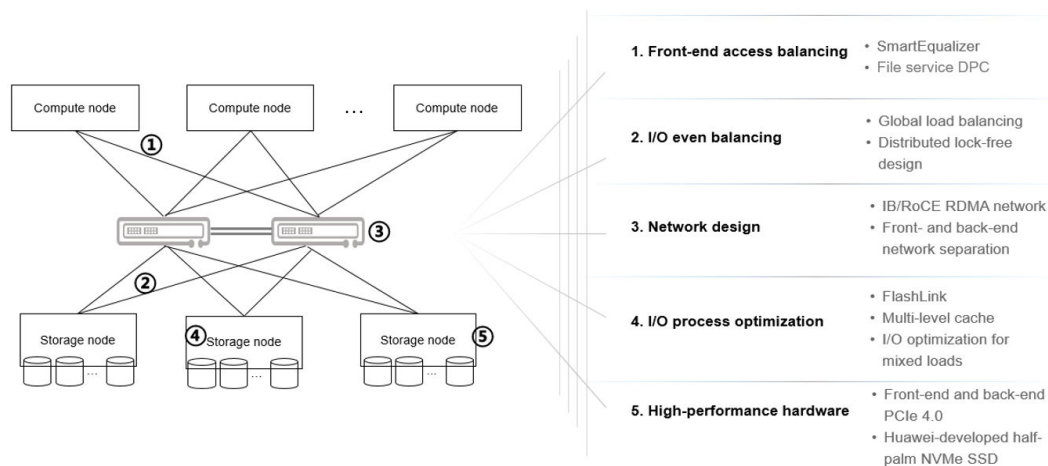


Table 6-1 Key performance design of Huawei OceanStor Pacific series storage system

Full I/O Process	Challenge	Key Design	Performance Design Principle
Front-end access balancing	- How to ensure access load balancing for NAS domain name access? - Only one node is connected to the traditional NAS client when the file system is mounted to the traditional NAS client. How to avoid the bottleneck of the access node?	- NAS access load balancing - DPC access	- The NAS service of OceanStor Pacific series provides domain name resolution and returns the IP addresses of storage nodes through SmartEqualizer. For details, see 4.4 SmartEqualizer (Load Balancing). - DPC access: One client can connect to multiple storage nodes, achieving I/O-level load balancing. For details, see 6.3 File Service Parallel Client.
I/O even balancing	- The processing capability of a single node is limited. How to achieve linear increase of distributed performance as the number of nodes increases? - How to balance the capacity of multiple nodes in a distributed environment?	Global load balancing	- Unstructured data is globally distributed by directory. - For structured data, LUNs are divided and distributed by slice. - Dynamic intelligent partitioning and static disk selection ensure balanced back-end capacity. For details, see 3.2 Global Load Balancing.

Full I/O Process	Challenge	Key Design	Performance Design Principle
Network design	How to reduce network data access and save bandwidth to improve performance?	• The front-end storage network is separated from the back-end storage network, and RDMA is supported. • DIO data passthrough. Data is written to the persistence layer through the front-end storage network after EC calculation on the DPC side.	• The RDMA network is supported. The front-end and back-end storage networks are separated. I/Os are rebuilt and balanced on the back-end network, which does not affect the front-end I/O performance. For details, see 3.1.2 Logical Network Architecture and 6.2 RDMA High-Speed Network. • In DPC access mode, data is written to the persistence layer from the front-end network after EC calculation on the DPC side. For details, see 6.3 File Service Parallel Client.

Full I/O Process	Challenge	Key Design	Performance Design Principle
I/O process optimization	- How to make full use of multi-CPU, multi-core, and all-flash media capabilities? - How to reduce I/Os delivered to disks? - How to optimize mixed large and small I/Os?	- FlashLink Intelligent multi-core technology ROW full-stripe write Intelligent stripe aggregation End-to-end I/O priority - Multi-level cache - Distributed lock-free design - Mixed I/O load optimization	- CPUs are partitioned based on services to reduce mutual interference between services. I/Os of different types are marked with priorities. Based on the priorities, the system controls CPU resources, memory resources, and queues. For details, see 6.1 FlashLink. - End-to-end multi-level data cache improves the read hit ratio. In addition, data is intelligently aggregated and written to disks. Data is written to disks as full-stripe as possible. For details, see 3.3 Multi-Level Cache Acceleration. - The distributed lock-free design eliminates the distributed serialization overhead caused by distributed locks. For details, see 6.4 Distributed Lock-Free Design. - A large data block is written from a DPC compute node to a persistence node through one network communication. Small data blocks are forwarded to the cache of the home node and aggregated into large data blocks

Full I/O Process	Challenge	Key Design	Performance Design Principle
			and then written to disks. For details, see 3.4.1 Key I/O Processes of Unstructured Data Storage Services .
High-performance hardware	How to maximize the performance per unit area?	- Front-end and back-end PCIe 4.0 - Huawei-developed half-palm NVMe SSD	For details, see 2.2 High-Performance Hardware Design .

- [6.1 FlashLink](#)
- [6.2 RDMA High-Speed Network](#)
- [6.3 File Service Parallel Client](#)
- [6.4 Distributed Lock-Free Design](#)
- [6.5 Mixed I/O Load Optimization](#)
- [6.6 GPU Data Read/Write Acceleration](#)
- [6.7 AI Cache](#)

6.1 FlashLink

The core of FlashLink is to maximize the performance of flash memory by using a series of optimization technologies for flash media. For flash media, FlashLink of Huawei OceanStor Pacific series uses the following key technologies to ensure high performance: data-control separation, intelligent multi-core technology, ROW full-stripe write, intelligent stripe aggregation, and end-to-end I/O priority.

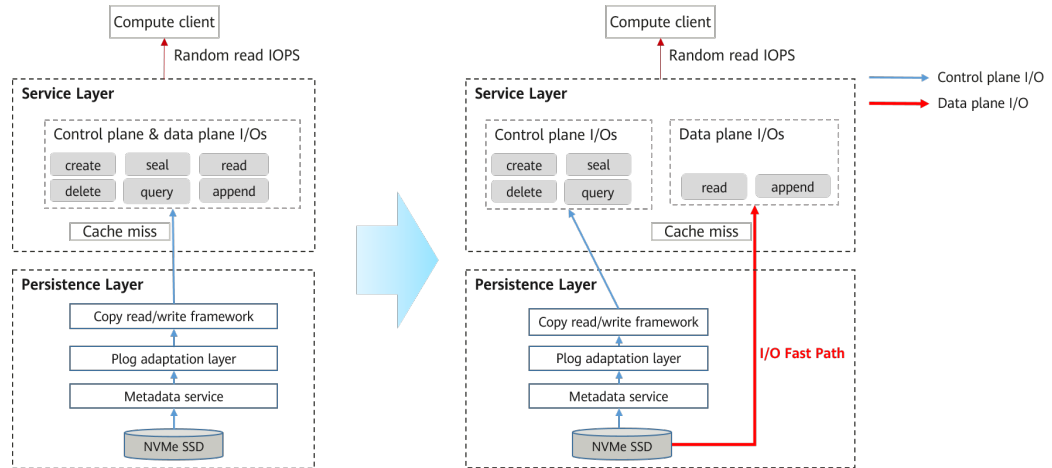
6.1.1 Data-Control Separation

In random read IOPS scenarios, the read cache hit ratio is low and the performance mainly depends on the disk performance. For this I/O model, OceanStor Pacific series optimizes the read and write efficiency of SSDs, and separates the I/O path of the data plane from that of the control plane, which is called data-control separation.

The storage software provides northbound storage semantic interfaces, including data plane I/O interfaces (append and read) and control plane I/O interfaces (create, delete, seal, and query). OceanStor Pacific series separates the I/O path of the data plane from that of the control plane to form two independent I/O scheduling frameworks, reducing the function call stacks of I/O operation

interfaces by 50% and the latency consumption and CPU usage of the storage software stack by 50%. The following figure shows the working principles.

Figure 6-2 Data-control separation diagram



6.1.2 Intelligent Multi-Core Technology

OceanStor Pacific series supports x86 and Kunpeng CPU platforms. Each node supports a maximum of two CPU sockets. If the Kunpeng CPU platform is used, each Kunpeng 920 CPU supports a maximum of 64 physical cores. On the x86 platform, a CPU with hyper-threading enabled supports multiple logical cores. The intelligent multi-core technology can implement better linear CPU scalability and maximize the CPU processing capability.

OceanStor Pacific series divides CPUs into different partitions based on service characteristics. Different types of services are allocated to different CPU partitions. In addition, with the non-uniform memory access (NUMA, which is an architecture model for memory access from multiple CPUs)-aware design, processes running in each CPU partition can access the nearest physical memory connected to a socket, to avoid long latency caused by cross-CPU access. In this way, overheads caused by the access latency of CPUs can be reduced, which greatly improves software running efficiency.

Thanks to CPU partitioning and dynamic binding of service threads to CPU partitions, different services run on different cores, preventing CPU contention and conflicts between service groups. In a multi-core system, this can reduce the overheads of common atomic variable and spinlock mechanisms and enhance the linear scalability of the entire system.

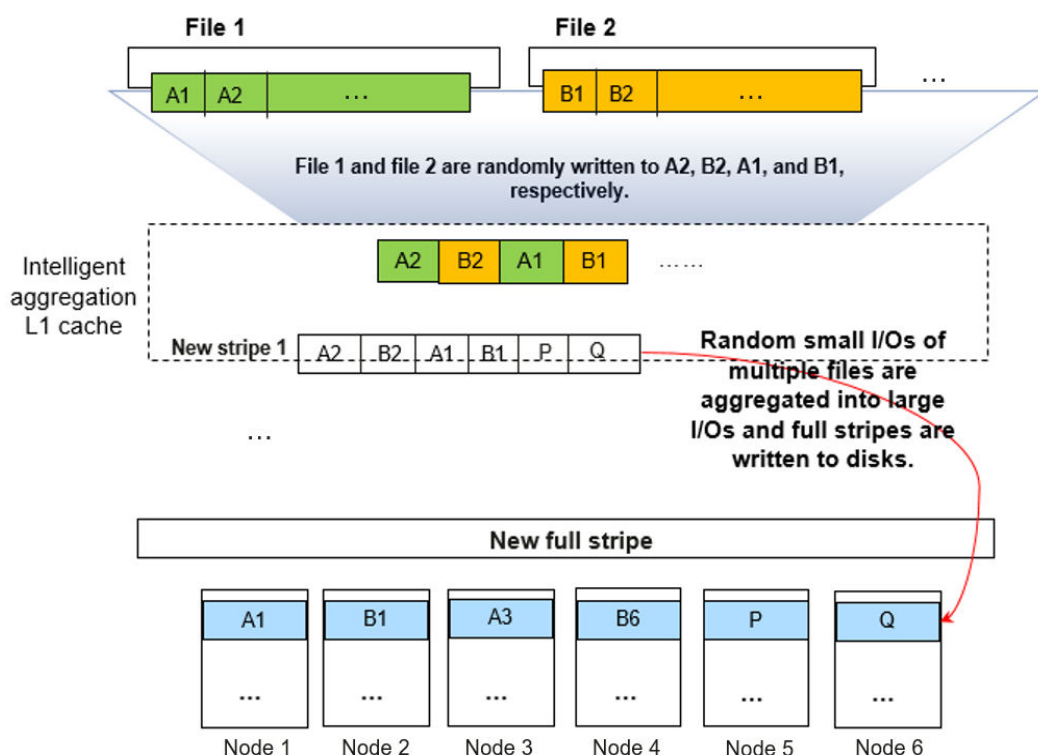
6.1.3 ROW Full-Stripe Write

Flash chips on SSDs can be erased for a limited number of times. In traditional RAID overwrite (write in place) mode, hot data on an SSD is continuously rewritten, and its mapping flash chips wear out quickly. Huawei OceanStor Pacific series storage with all-flash configuration uses redirect on write (ROW) full-stripe write for both new data writes and old data rewrites. It allocates a new flash chip for each write, balancing the number of erase times of all flash chips. This greatly reduces the overhead on node CPUs and read/write loads on SSDs in a write process.

6.1.4 Intelligent Stripe Aggregation

OceanStor Pacific series supports EC redundancy protection. To provide high space utilization, the product supports efficient large-ratio EC ratios, for example, 20+2. When the EC ratio is large, random small I/Os cannot be written to disks in full stripes. The traditional processing mode generates extra write penalty I/Os, resulting in performance deterioration. To solve this problem, OceanStor Pacific series implements the intelligent aggregation technology for small I/Os based on the "Plog append only" semantic design. A Plog is a set of physical addresses managed by the persistence layer based on a fixed size. It provides the EC redundancy protection capability and accesses the LBA address on a specific physical disk in Plog ID + Offset mode. When data is written, the physical space is allocated and the mapping relationship between the logical address of Plog ID + Offset and the physical LBA address is established. Take the file service as an example. Random small I/Os of multiple files are written to the L1 cache and aggregated into large I/Os. Full-stripe I/Os are written to Plogs to reduce write penalties and improve system performance.

Figure 6-3 Intelligent stripe aggregation

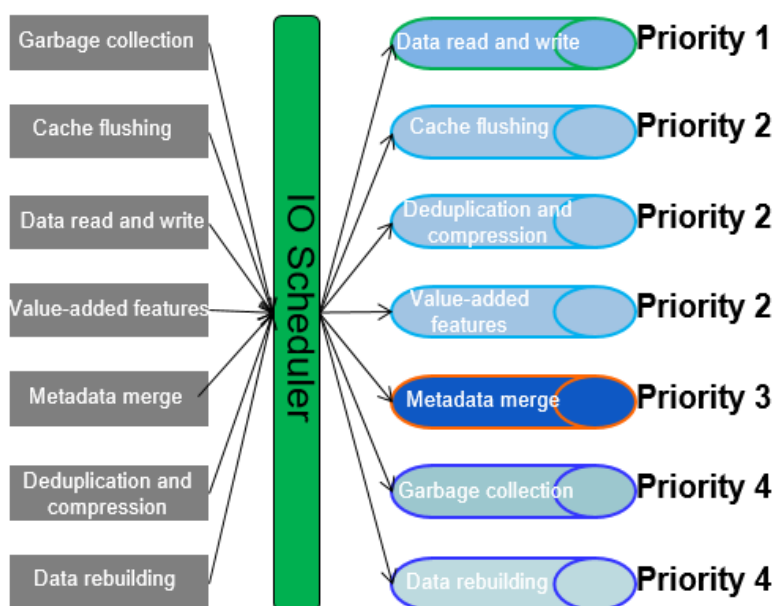


6.1.5 End-to-End I/O Priority

To ensure stable latency, OceanStor Pacific series marks I/Os of different types with priorities. Based on the priorities, the system controls CPU resources, memory resources, and queue scheduling to ensure end-to-end priority assurance and high-priority response to host data read and write I/Os. Different priorities are assigned to background I/Os based on their types to ensure the latency of host data read and write I/Os.

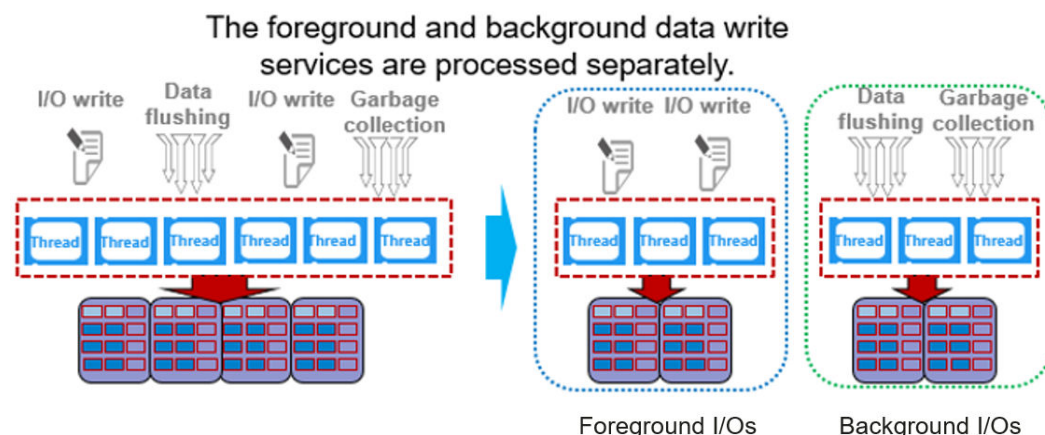
For example, when a host read/write request is received, I/Os with higher priorities will be processed first to ensure stable latency and quick response to host requests. System background I/Os are classified into various types: cache flushing, metadata merge, advanced features (remote replication and background copy), data garbage collection, deduplication and compression, and data rebuilding. The priorities of these I/Os are in descending order.

Figure 6-4 I/O priorities



To prevent background services from affecting host write services, the foreground and background services are processed separately. The foreground and background data services have independent thread resources, I/O concurrency resources, and storage space, reducing the impact of background disk flushing, data GC, and metadata merge services on host write services, ensuring the stable latency of the system.

Figure 6-5 Foreground and background services



6.2 RDMA High-Speed Network

A single scale-out storage cluster usually consists of dozens or even thousands of storage nodes. Data is evenly distributed to different nodes through the storage network, which poses high requirements on the performance and reliability of the network. All nodes of the OceanStor Pacific series file service are interconnected through an RDMA network which enables low-latency communication. This is one of the bases for high IOPS, large bandwidth, and low latency of OceanStor Pacific series.

Huawei OceanStor Pacific series supports IB and RoCE networks (IB and RoCE for the file service, and RoCE for the HDFS service) and can use the RDMA protocol for I/O data exchange. Compared with the TCP protocol, the RDMA protocol has no TCP protocol stack processing overhead and can provide a faster response.

Huawei-developed network chips support RoCE, RDMA, and IP address-based access. When supporting different types of services such as bandwidth and IOPS, the chips intelligently adjust network parameters to provide better congestion control and greatly reduce the latency of data exchange between different storage nodes. The Huawei-developed switch AI Fabric technology is used to further reduce the system access latency.

To reduce the number of network hops and improve the end-to-end performance, the OceanStor Pacific series unstructured services allow request access nodes to read and write large I/O data from persistent nodes, and the OceanStor Pacific series structured service can write large I/O data to persistent nodes from the home node without passing through the L1 cache.

- NFS over RDMA

The NFS protocol depends on the RPC protocol, and the lower layer of the RPC protocol is the transport layer. The traditional NFS transport layer uses the TCP or UDP protocol. NFS over RDMA depends on the RDMA technology to implement data passthrough between NFS clients and storage devices. Compared with TCP transmission, NFS over RDMA effectively reduces network latency, lowers CPU loads on clients and servers, and improves NFS access performance.

6.3 File Service Parallel Client

High-performance computing (HPC) is to interconnect a large number of servers through high-speed networks to form a computer cluster. HPC works with high-performance storage to solve the most complex scientific computing problems in scientific research and industry. After years of development, HPC has been used in education and scientific research, life science, manufacturing simulation, weather forecast, and many other fields. With the development of IoT and intelligent technologies, HPC scenarios such as HPC, big data, and High Performance Data Analysis (HPDA) emerge. HPC further applies to fields such as self-driving, financial anti-fraud, and personalized healthcare.

In HPC scenarios, storage systems will face more complex I/O models. To cope with these performance challenges, OceanStor Pacific series file service launches the Distributed Parallel Client (DPC) to carry standard POSIX semantics and MPI-

IO semantics, improving the performance of a single stream and a single client and reducing the access latency. DPC runs on a compute node and functions as a storage client. It exchanges data with back-end storage nodes through network protocols. DPC is compatible with standard POSIX and MPI-IO semantics and provides parallel interfaces and an intelligent data cache algorithm, so that upper-layer applications can access storage space more intelligently. The following table lists the comparison between the DPC protocol and the traditional standard NFS protocol.

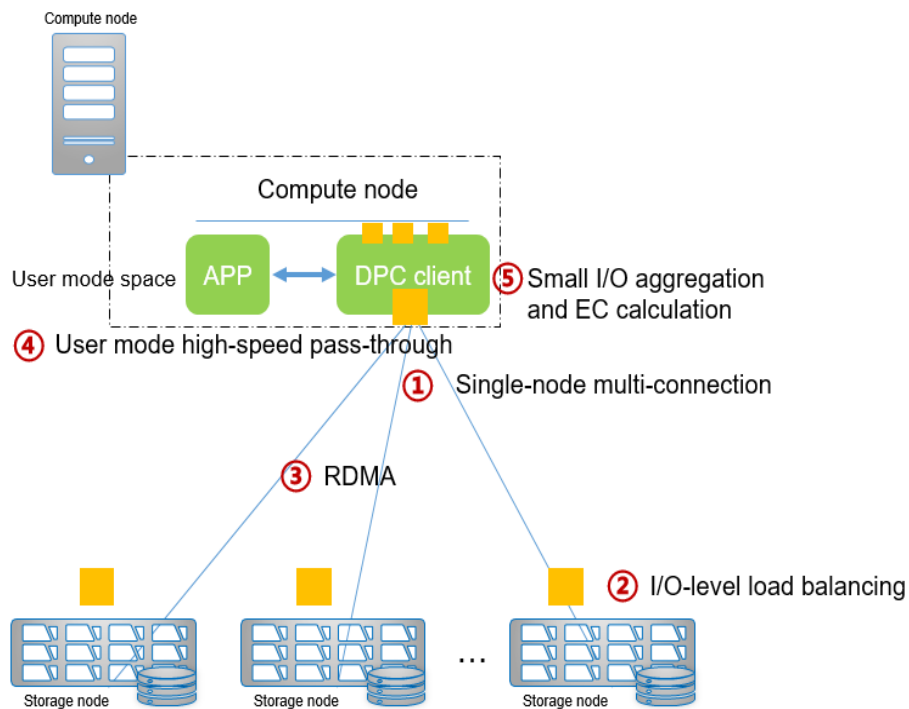
Table 6-2 Comparison between the DPC protocol and the traditional standard NFS protocol.

Item	NFS	DPC
Single-client connection	A single client can connect to only one storage node.	A single client can connect to multiple storage nodes.
Load balancing	Connection quantity balancing, which cannot ensure real-time load balancing of storage services.	I/O-level load balancing, which ensures real-time load balancing of storage services.
Data I/O path length	Data I/Os can be written only to a mounted storage node. To write data to another target storage node, you need to cache the data to a mounted storage node and then write the data to the target storage node.	Data I/Os can be directly written to target storage nodes.
Single-stream performance	The performance is limited by the minimum performance bottleneck of a single CPU core on the storage and computing sides.	There is no performance bottleneck on the storage side. The performance is limited only by the performance of a single CPU core on compute nodes.
Interface support	POSIX	POSIX and MPI-IO
Communication mode	TCP/IP	RDMA
NUMA affinity	Not supported	Supports NUMA affinity for compute applications, leveraging the ultimate performance of the multi-NUMA architecture.

In POSIX semantics, a compute node that runs the DPC process is connected to all storage nodes in the cluster over networks. I/Os delivered by applications are first

aggregated on the compute node, calculated by EC, and then written to the plog persistent node (persistence layer) by PlogClient. Compared with traditional NFS and SMB protocols, data delivered by the service layer is written to the persistence layer through the RDMA network. This reduces network forwarding times required for storage nodes by one and greatly decreases CPU usage, as shown in the following figure:

Figure 6-6 I/O on the DPC side

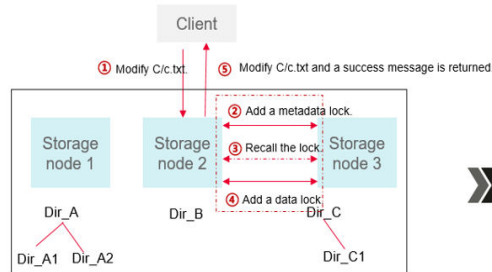


1. A single client is connected to multiple storage nodes, improving the single-stream and single-client bandwidth.
2. The problem that load balancing can be performed only during mounting based on the standard protocol is resolved to implement I/O-level load balancing.
3. The RDMA protocol is supported to achieve lower latency and lower CPU overhead.
4. High-speed passthrough between user-mode processes is implemented between DPC and applications to avoid multiple copies of data from the user mode to the kernel mode. Asynchronous I/Os can be directly completed in the kernel mode to ensure cross-node data consistency. In addition, the intelligent pre-fetch algorithm is used to automatically identify file I/O access rules and access rules of mass small files in directories, and efficiently pre-fetches file metadata and data without manual parameter optimization, meeting high-performance requirements of various HPC applications.
5. Random small I/Os are aggregated into large I/Os. After calculating the EC redundancy on the DPC side, the large I/Os are written to persistent nodes to improve data write performance.

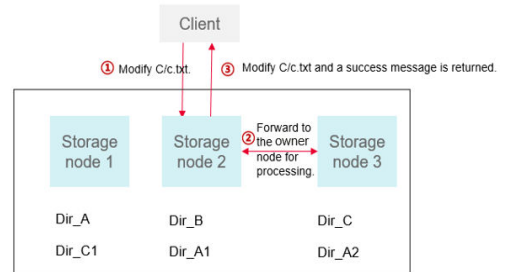
6.4 Distributed Lock-Free Design

Figure 6-7 Distributed lock-free design

Process of processing small I/Os based on distributed locks



Process of forwarding small I/Os to the owner node



The OceanStor Pacific series file service implements one-time distribution logic based on the directory partition granularity and introduces the ownership forwarding mechanism of the centralized architecture. Small I/O requests are processed by the owner node in a unified manner, eliminating the need for distributed locks. Small I/O operations require only one message interaction, greatly reducing CPU overheads and request latency.

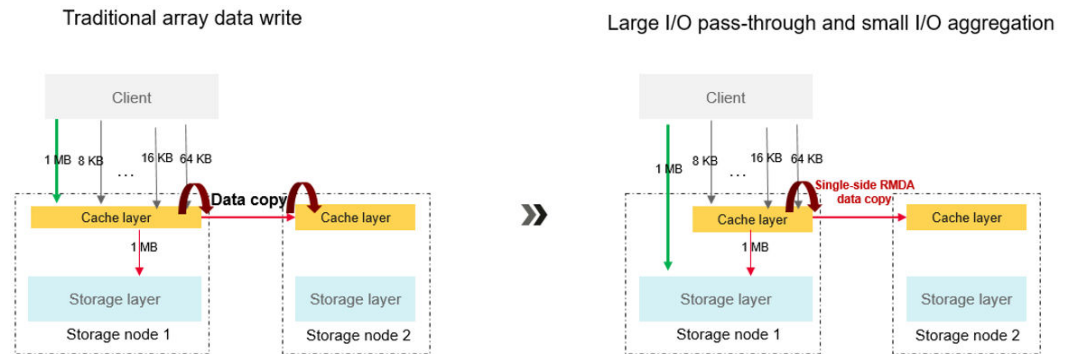
6.5 Mixed I/O Load Optimization

HPC has various service loads, including bandwidth for sequential large I/Os, high IOPS for random small I/Os, OPS for batch metadata operations, and parallel I/Os for accessing the same file at the same time. With the convergence of traditional HPC, big data, and intelligent technologies, heterogeneous computing, such as ARM, GPU, and FPGA, further improves the requirements on storage performance. How to meet the requirements of ultra-high bandwidth, ultra-high IOPS, and ultra-low latency at the same time is the biggest challenge for HPC storage performance.

OceanStor Pacific series file service solves the problem of coexistence of bandwidth and IOPS services through multi-dimensional innovation.

- The large I/O pass-through technology is used to reduce network amplification, the large-proportion EC technology is used to reduce disk bandwidth amplification, and the large-block sequential write technology based on the FlashLink technology is used to implement sequential disk I/O access, achieving ultra-high bandwidth performance.
- The multi-level intelligent cache and small- and medium-I/O aggregation technologies as well as the single-side RDMA technology are used to reduce latency caused by deep software stack and CPU computing performance bottleneck, achieving high IOPS and low latency.
- FlashLink's intelligent multi-core technology and end-to-end I/O priority scheduling technology are used to resolve the requirement conflict in scenarios where bandwidth and IOPS services coexist.

Figure 6-8 Large I/O pass-through and small I/O aggregation



When data is written to traditional storage arrays, all I/Os are written to the memory first. After node mirroring is complete, the I/Os are returned to services. For small I/Os, the path is the shortest and the latency is the lowest. However, for large I/O bandwidth-intensive services, the network bandwidth and memory bandwidth are doubled, and the service bandwidth is only half of the network bandwidth and memory bandwidth. For mixed loads, large I/Os are passed through to disks to reduce network and memory bandwidth amplification and achieve the highest bandwidth performance. Small I/O write cache aggregation implements mirroring and response, achieving the lowest latency. However, the number of CPU scheduling times is increased, which further aggravates the CPU performance bottleneck. Based on this, OceanStor Pacific series file systems implement single-side data copy in RDMA mode, which occupies only CPU resources of the main storage node. The total CPU overhead is reduced by 20% to 30%. The bandwidth and IOPS can be optimal in mixed load scenarios.

6.6 GPU Data Read/Write Acceleration

Big data, HPC, AI, and ML applications, such as image classification, video analysis, speech recognition, and natural language processing, offload computing to GPUs to complete tasks more quickly, which poses high requirements on GPU performance.

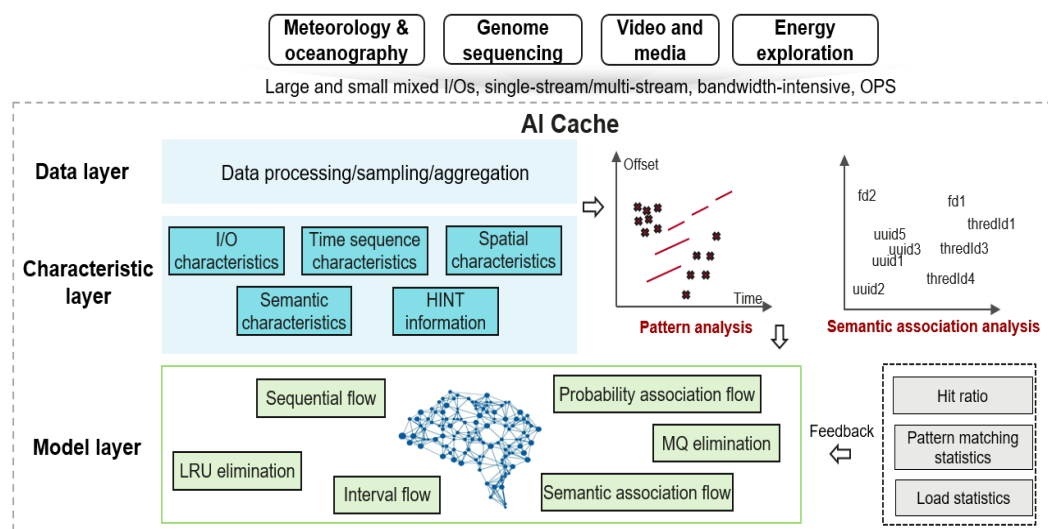
Depending on the NFS over RDMA feature, OceanStor Pacific series uses the GPU Direct function to accelerate GPU data read/write in passthrough mode. That is, GPUs can directly read data from or write data to storage systems at a high speed without the assistance of CPUs. This increases the storage system bandwidth, reduces CPU latency, and improves GPU data read/write performance.

6.7 AI Cache

HPC, big data, and AI scenarios require high bandwidth, IOPS, and OPS, and also face challenges such as file access disorder and streaming causes by complex factors such as multi-load, multi-protocol, and limited resources. As a result, the access latency is high, which severely affects data processing efficiency.

OceanStor Pacific series provides the AI Cache technology to implement intelligent workload awareness. By learning and mining file I/O characteristics, file semantic information, and directory operation rules, the system effectively associates user behavior with storage data characteristics, achieving optimal prefetch and

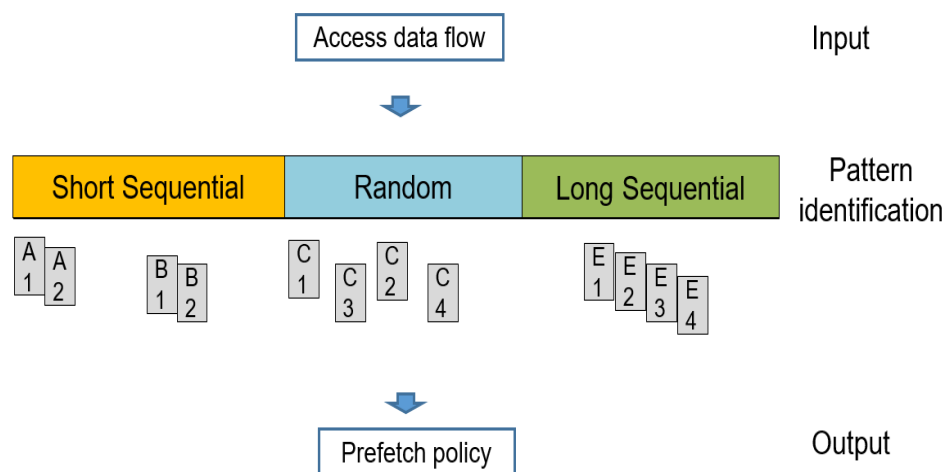
elimination within and across files. In addition, the performance of large files and small I/Os is improved by up to 15 times, and the performance of small files is improved by 2 times.



AI Cache uses the multi-dimensional intelligent file semantic prefetch algorithm to construct feature profiles of adjacent connections and interactive connections with features such as space-time, file semantics, and directory operation behavior as points and access as lines. The built-in streaming machine learning model has advantages in computing power, memory resource utilization, and high precision, and therefore can implement online policy optimization and I/O access prediction in adaptive scenarios.

- **Application-aware pattern identification algorithm**

According to the spatial characteristics, time sequence characteristics, and application information of file access, the file access sequence, interval, and mode (random or not) can be identified. The optimal prefetch length is calculated adaptively by reusing the distance correlation model to identify the patterns generated by different applications in the same file and solve the problem of disorder and streaming in complex service models.



- **Semantic-aware prefetch algorithm**

File name semantic feature mining: By mining file name features such as characters, numbers, extensions, and dates, the regression algorithm is used

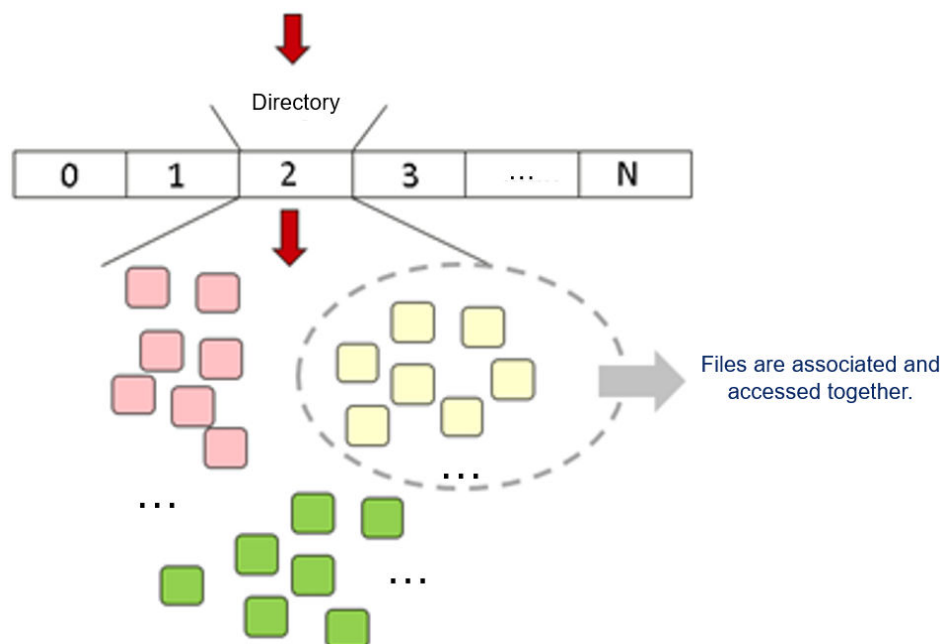
to learn and infer multi-dimensional prefetch policies. The algorithm supports prefetching of linear number change rules in file names and prediction of associated extensions. As shown in the following figure, the number in the file name in the file access flow changes in ascending order. After learning the ascending rule of the number in the file name online, the system predicts the next file that may be accessed.



Directory operation behavior identification: The directory operation acceleration algorithm is adopted to learn I/O access information to identify operation behavior and recommend prefetch sequences in directory order and lexicographic order, implementing prefetch at the directory granularity and directory copy acceleration.

- **Cross-file association prediction algorithm**

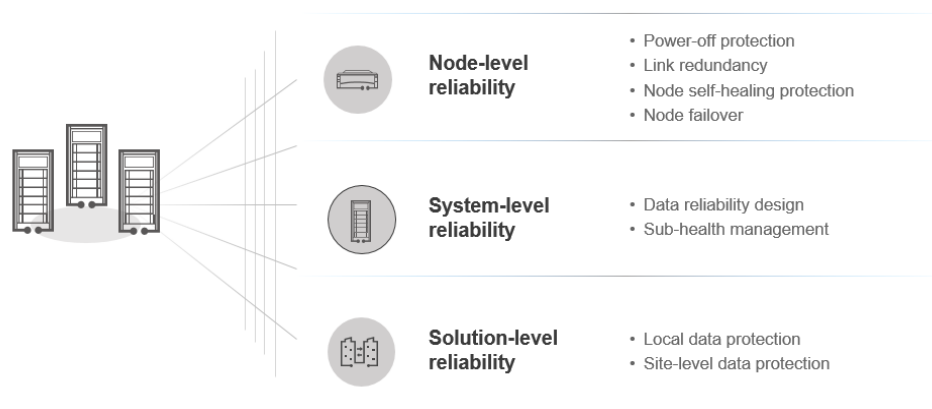
To cope with scenarios where file names and directory operations are irregular, a model used to associate probabilities and read/write in file access is built. In addition, a compact prediction tree is generated with low resource overhead to implement cross-file time and probability association prefetch. As shown in the following figure, the inter-file probability model is learned and iterated incrementally. After a file is accessed, the file with the most associated time or probability will be accessed later.



7 System Reliability Design

OceanStor Pacific series provides 99.999% system reliability at node and system levels. Together with DR solutions, the system reliability can reach 99.9999%. OceanStor Pacific series implements flexible security layout and redundancy of data fault domains. In addition, end-to-end data integrity protection and data protection in various fault scenarios are designed to implement high-reliability storage and service processing of data information.

Figure 7-1 End-to-end reliability guarantee of OceanStor Pacific series



[7.1 Module-Level Reliability](#)

[7.2 Node-Level Reliability Design](#)

[7.3 System-Level Reliability](#)

[7.4 Solution-Level Reliability Design](#)

7.1 Module-Level Reliability

7.1.1 Disk Reliability

A storage system contains a large number of disks that carry mission-critical user data. Therefore, promptly detecting and efficiently handling disk faults is a challenge that faces all storage systems. To improve the fault tolerance capability of disks, Huawei OceanStor Pacific series provides a series of fault tolerance designs.

- Intelligent scanning of bad sectors

A disk having bad sectors does not automatically report the fault. Bad sectors can be detected only when data is read from or written to them, or during data verification. Huawei OceanStor Pacific series periodically scans disks without affecting services and disk reliability so that bad sectors can be detected within a short time. The system reads redundant data on other nodes and writes the data to the bad sectors for self-healing.

- Disk fault prediction

After disks run for a long time, their components may age, and the fault rate will increase over time. Huawei OceanStor Pacific series periodically checks the Self Monitoring Analysis and Reporting Technology (SMART) information of disks, monitors key indicators that may cause disk faults, and predicts disk faults by analyzing the change trend of key indicators. For all types of disks (main storage, cache, system, and metadata disks), an alarm is reported when the system predicts that a disk is about to fail.

- Slow disk detection and isolation

Disk aging, firmware defects, temperature exceptions, and vibration exceptions may cause slower I/O responses than those in normal operations. When some disks respond slowly, the entire storage system may slow down. As a result, host applications in the storage system freeze or are interrupted. When service I/Os are running, Huawei OceanStor Pacific series collects statistics about the average response time of all I/Os delivered to disks within a specified period of time in a real-time manner and uses the intelligent clustering algorithm (a machine learning algorithm) to identify slow disks. For main storage or cache disks, after identifying a slow disk, the system isolates it and triggers data rebuilding. For system or metadata disks, after identifying a slow disk, the system reports an alarm.

- Slow I/O processing

When an exception occurs on a disk or link, the latency of certain I/Os on the disk may be abnormal. As a result, the response time of host I/Os is affected and host service performance fluctuates. Huawei OceanStor Pacific series monitors the response time of each I/O delivered by host services. If the response time exceeds the preset threshold, for a main storage or cache disk, data is obtained from other copies or in EC degraded read mode in read operations, and another disk is selected and redundant write is used to quickly respond to hosts in write operations. For a system or metadata disk, the system reports an alarm.

- Disk error handling

Huawei OceanStor Pacific series proactively identifies disk faults, such as write protection (WP), command abortion (ABRT), and disk failure (DF), during I/O processing. After a main storage or cache disk is faulty, the system reports an alarm and triggers data rebuilding. After a system or metadata disk is faulty, the system reports an alarm.

- Disk roaming

Disk information is recorded inside disks and checked when disks are inserted. Huawei OceanStor Pacific series allow disks to be exchanged between any slots on the same storage node without affecting disk identification and data read/write. Specifically, Huawei OceanStor Pacific 9550 is a 5 U high-density and large-capacity device. 120 disks in a single chassis are managed by four expanders, and only disk roaming within the same expander is supported.

7.1.2 HSSD Reliability

With more than 10 years of accumulated development experience and an in-depth understanding of numerous customer application scenarios, Huawei offers a comprehensive reliability design and overall protection for disk controllers, NAND flash, algorithms, and disks. HSSDs provide reliability technologies such as data protection, die failure management, bad block management, background inspection, and trim mode. For details, see *Huawei OceanStor Pacific Series Reliability Technical White Paper*.

7.1.3 NIC Reliability

Huawei OceanStor Pacific series uses the bond technology to aggregate multiple network ports to implement fault redundancy and load balancing. Currently, active-backup (bond1), balance-xor (bond2), and 802.3ad (bond4) bonding modes are supported.

In addition, Huawei OceanStor Pacific series performs monitoring and failover in NIC sub-health scenarios, including but not limited to:

- Intermittent disconnection of a network port
- Incorrect network port negotiation rate
- Incorrect negotiation rate of the NIC PCIe bus
- Packet loss or packet error over a network port
- Long network latency

7.1.4 Memory Reliability

As the memory capacity and memory speed increase, memory faults occur at a higher probability. After a memory fault occurs, the faulty memory area is isolated and recovered at the system layer. This improves the system fault-tolerance capability and reduces the fault rate.

When the system is running properly, the memory controller periodically inspects the memory. After a correctable error (CE) is detected, the ECC algorithm is used to restore data and re-write the restored data to the memory again, preventing errors from being accumulated. After an uncorrectable error (UCE) is detected, the system uses the online page isolation technology to isolate the faulty memory area and nodes continue running.

During data reads and writes, if a single-bit error occurs in the memory, the ECC algorithm is used to repair the faulty memory area and then the memory area continues providing services. After a memory hardware fault occurs, it may be detected and corrected repeatedly. As a result, a CE interrupt continuously occupies CPU resources. To prevent this problem, CPUs collect statistics on CE

interrupts. When the number of CE interrupts exceeds the threshold, the operating system isolates the memory areas where CEs frequently occur by page. If the faulty memory area has not been allocated, the operating system and memory management module do not allocate the memory. If the faulty memory area is in use, the memory management module does not allocate the memory after the memory is released and reclaimed, avoiding frequent ECCs.

During data reads and writes, if a user-mode process accesses the memory area where a UCE occurs, the process will be stopped and restarted. In addition, the online page isolation function is supported for memory UCEs. When a UCE occurs in the memory, the address of the faulty memory area is recorded in non-volatile storage media. After the process is restarted, the faulty memory page is isolated based on the recorded UCE address to prevent the error page from being accessed again and avoid repeated process exceptions.

7.2 Node-Level Reliability Design

7.2.1 Power Failure Protection

Nodes may be powered off while the system is running. Huawei OceanStor Pacific series uses non-volatile media to store metadata and cache data, preventing metadata and user data from being lost.

OceanStor Pacific series can use SSDs as non-volatile media. On a storage device where the OceanStor Pacific series structured service uses SSDs as cache disks, the system writes metadata and cache data to non-volatile media during program running. After a node is powered off and restarted, the system recovers the metadata and cache data in non-volatile media automatically. Alternatively, on a storage device where OceanStor Pacific series unstructured services use SSDs as power backup disks, the system writes metadata and cache data to the memory during program running. After a node is powered off, the system writes the metadata and cache data to the power backup SSDs using the storage device's backup battery. After the node is restarted, the system recovers the metadata and cache data in the power backup disks automatically.

7.2.2 Link Reliability

Huawei OceanStor Pacific series leverages the link aggregation technology to implement transmission link redundancy and perform link switchover or isolation when a link becomes faulty or sub-healthy to ensure service continuity. Link aggregation combines multiple physical links into a single logical link for applications. When a physical link becomes faulty, communication is switched to other links.

Clusters support network isolation. The management plane and the storage plane are physically separated (different VLANs are divided) and served by different components. Therefore, services are not affected if the management plane is faulty. Congestion on the service plane does not affect the management plane. Each functional plane can independently detect, rectify, and isolate faults without affecting each other. The following network configuration modes are supported:

- Two networks: front-end storage network and management network

- Three networks: front-end storage network, back-end storage network, and management network

When the cluster uses Ethernet, IB, or RoCE networks, link reliability is ensured through the following methods:

1. For Ethernet networks, the front- or back-end storage network of each node is configured with two Ethernet ports connecting to two switches, respectively. The two network ports use the built-in bonding driver of the operating system to perform link aggregation. The following bonding modes are supported:

- bond1: active-backup
- bond2: balance-xor (active-active)
- bond4: 802.3ad (active-active)

The system supports proactive switchover of sub-healthy network ports only when the bond1 mode (active-backup) is configured. In this mode, an aggregation network port corresponds to multiple physical network ports, including one active network port and other backup network ports. After the active network port is faulty or sub-healthy, the system automatically switches the active network port to a backup one.

2. For IB networks, the front- or back-end storage network of each node is configured with two IB network ports connecting to two switches, respectively. The two IPoIB network ports of the storage network on a node use the built-in bonding driver of the operating system to perform link aggregation. System services (I/O flows) are not restricted by the bond configurations of the IPoIB network ports and can communicate with each other through two RDMA devices at the same time. An active-active bonding mode is used. After an IB network port is faulty or sub-healthy, the system isolates the corresponding RDMA device. For the corresponding IPoIB aggregation network port, if the faulty network port is the active one, the active network port is switched to a backup network port.
3. For RoCE networks, the front- or back-end storage network of each node is configured with two Ethernet ports connecting to two switches, respectively. The system uses the built-in bonding driver and RoCE NIC driver (supporting NIC firmware) of the operating system to perform link aggregation. The following bonding modes are supported:
 - bond1: active-backup
 - bond2: balance-xor (active-active) (The RDMA route selection mode is determined by an NIC.)
 - bond4: 802.3ad (active-active) (The RDMA route selection mode is determined by an NIC.)

7.2.3 Node Self-Healing Protection

7.2.3.1 High Temperature Protection

Huawei OceanStor Pacific 9950/9550 storage system monitors the ambient temperature and key temperature points of the system in real time. Based on the information about each monitoring point, the system dynamically adjusts the fan speed to ensure that the system and components work within the normal temperature range.

When the ambient temperature increases or a component is abnormal, the fan speed cannot be adjusted to ensure that the system or component works within the normal range. In this case, two thresholds are set. When the temperature exceeds the level-1 threshold, an overtemperature alarm is generated. If the temperature keeps increasing and exceeds the level-2 threshold, the storage system is powered off to prevent component damage caused by continuous high temperature.

Huawei OceanStor Pacific 9920/9520/9540 series storage system monitors the temperature of disks in real time. When the temperature exceeds the level-2 threshold, disk access is stopped to prevent disk damage caused by continuous high temperature.

7.2.3.2 Hardware Self-Healing Protection

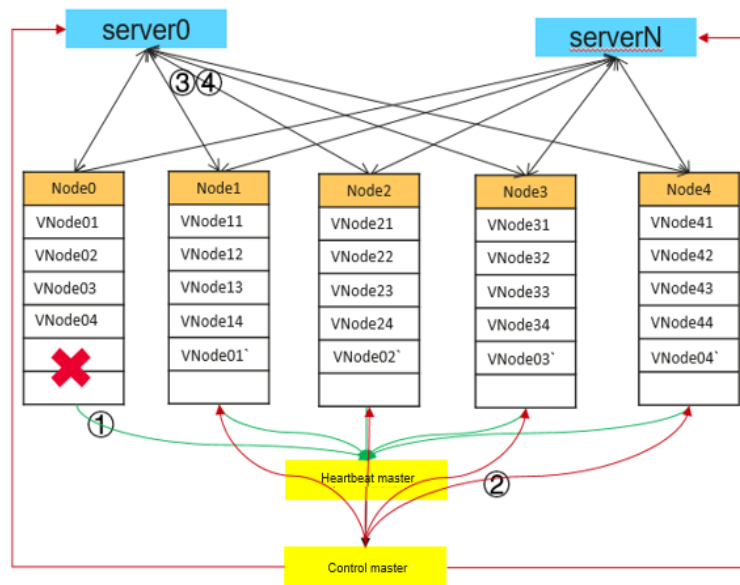
A scheduling exception indicates that a thread cannot be executed properly or a CPU cannot work properly due to a software bug or hardware exception. Huawei OceanStor Pacific 9550/9950 storage system has a dedicated chip (watchdog chip CPLD) outside the CPU to guard against CPU suspension. When detecting that the CPU is suspended, the system automatically resets nodes to ensure the redundancy of the storage system.

7.2.4 Node Failover

After a node is faulty (for example, the operating system is reset or a CPU is faulty), the fault can be quickly detected and services on the node are switched to other normal nodes. This ensures that the fault does not affect the availability of the storage system within the designed redundancy range. In addition, after the fault recovers and the node goes online again, services on other nodes are automatically switched back to the node to restore system availability and provide higher system performance and more storage space.

Huawei OceanStor Pacific series uses the heartbeat mechanism to detect node faults. Multiple mechanisms, such as the multi-node and multi-module concurrent service takeover, delayed loading, batch loading, and quick memory index rebuilding, are used to reduce the node failover duration and shorten the I/O suspension time.

Figure 7-2 Fast switchover and concurrent service takeover upon a node fault



1. Process fault detection and notification (1): notifies a process fault upon heartbeat disconnection.
2. Concurrent takeover of storage nodes (2): At the software layer, each physical node is logically divided into four vNodes. In the event of a physical node fault, services on the faulty node are taken over by other normal physical nodes at a granularity of vNode.
3. Fast DPC switchover on compute nodes (3 and 4): The system proactively notifies a DPC of a partition view change, and the DPC switches services to a new storage node based on the view.
4. Fast standard client switchover on compute nodes (4): Standard clients support IP address failover. The IP address of a faulty node can be automatically switched to a new storage node.

7.3 System-Level Reliability

7.3.1 Data Reliability Design

7.3.1.1 Data Redundancy Protection

Huawei OceanStor Pacific series provides cross-node or cross-cabinet data protection capabilities. When multiple disks, nodes, or cabinets are faulty, it can continue providing services.

7.3.1.1.1 Data Security Layout Policy

Huawei OceanStor Pacific series supports flexible data layout policies, including vNode-level, node-level, and cabinet-level security, that is, cross-vNode, cross-node, and cross-cabinet security. When multiple disks, vNodes, nodes, or cabinets are faulty, the storage system can still provide services. The unstructured services

support vNode-level (EC) and node-level (EC) security. The structured service supports vNode-level (EC), node-level (EC or multi-copy), and cabinet-level (EC or multi-copy) security. In particular, the structured service supports online conversion of node-level to cabinet-level security. The unstructured services support online conversion of vNode-level to node-level security by adding nodes.

- vNode-level security

Based on the dedicated OceanStor Pacific 9550/9546 hardware architecture of two nodes per chassis, a single node is logically divided into multiple vNodes. The system distributes original and redundant data on different vNodes. If the number of faulty vNodes is less than or equal to the number of redundant vNodes, data can be automatically recovered, ensuring that services are not interrupted and data is not lost. On a node in a chassis, even if the number of faulty vNodes is greater than the number of redundant vNodes (that is, a single node is faulty), the fast node switchover technology can still be used to switch data disks and services to the peer node. In this way, services are not interrupted, data is not lost, and reliability is not degraded.

- Node-level security

The system distributes original and redundant data on different nodes. If the number of faulty nodes is less than or equal to the number of redundant nodes, data can be automatically recovered, ensuring that services are not interrupted and data is not lost.

In EC mode, data and parity fragments are distributed on different nodes. For example, if the 4+2 EC scheme is configured for a storage pool containing eight nodes, the system can still provide services when two nodes become faulty.

- Cabinet-level security

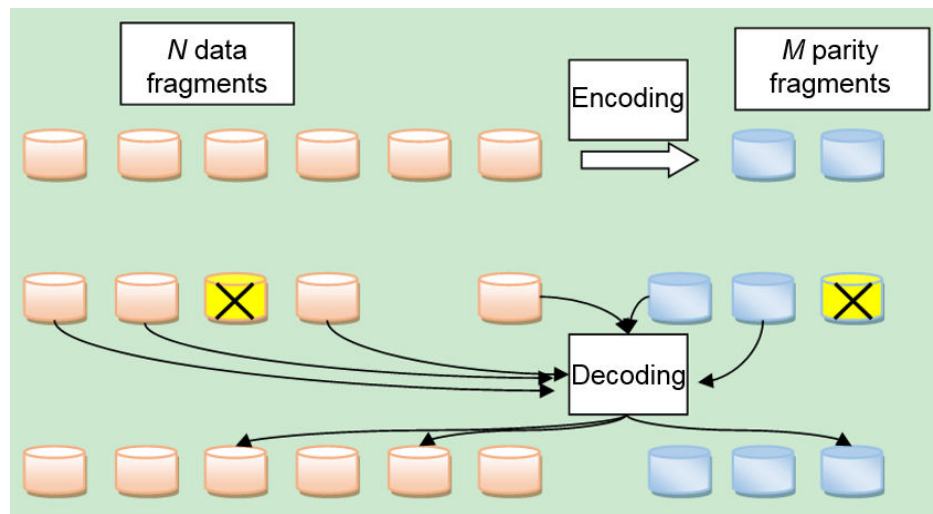
The structured service distributes original and redundant data on different cabinets. If the number of faulty cabinets is less than or equal to the number of redundant cabinets, data can be automatically recovered, ensuring that services are not interrupted and data is not lost.

In EC mode, data and parity fragments are distributed in different cabinets. For example, if the 4+2 EC scheme is configured for a storage pool containing eight cabinets, the system can still provide services when two cabinets become faulty.

7.3.1.1.2 Erasure Coding (EC)

Huawei OceanStor Pacific series uses the EC mechanism to ensure data reliability. EC uses various encoding methods to balance storage space utilization and data reliability. Data written into an OceanStor Pacific series storage system is divided into fixed-size data fragments. Every N data fragments are calculated to generate M parity fragments. The N and M fragments form a stripe and are written into the system. In case a fault occurs in the system, as long as the number of lost fragments in a stripe does not exceed M , data can be read and written properly. Lost strips can be restored from the remaining strips using a data restoration algorithm. The space utilization of the system is about $N/(N + M)$. M determines the data reliability. A larger value of M brings higher reliability.

Figure 7-3 Huawei OceanStor Pacific series storage system EC



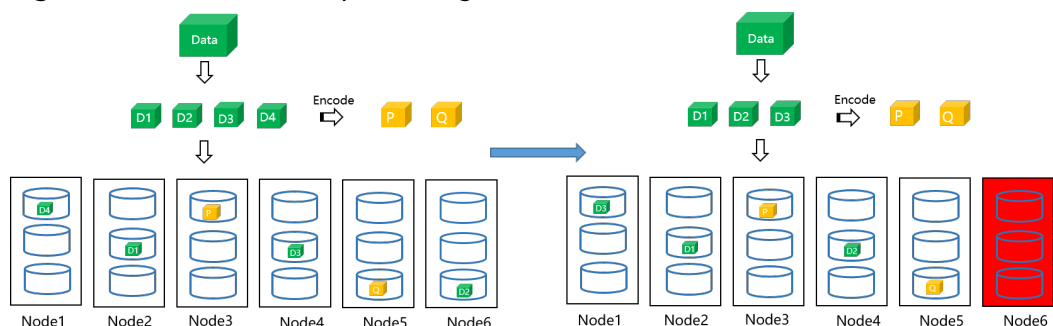
Huawei OceanStor Pacific series uses the low density erasure coding (LDEC) algorithm, which is a family of MDS array codes based on XOR and Galois field multiplication. The minimum encoding granularity is 512 bytes. The algorithm supports Intel instruction acceleration and various mainstream EC ratios. When the number of nodes is less than $M + N$, the $M+N:1$ EC scheme can be configured and multiple strips in the same stripe may be stored on the same node. In this case, even if M is 2, 3, or 4, only one node is allowed to be faulty, resulting in reliability deterioration.

For details about EC, see the *Huawei OceanStor Pacific Series Elastic EC Technical White Paper*.

7.3.1.1.3 Never Degraded Data Writes

Traditional RAID technology degrades write operations when a RAID member disk is faulty to ensure I/O continuity, and only records operation logs for damaged disks, which lowers the reliability. Unlike RAID technology, Huawei OceanStor Pacific series EC guarantees I/O continuity and provides the same reliability through reduction when a node is faulty.

Figure 7-4 EC reduction upon a single-node fault



As shown in the figure, the 4+2 EC scheme is configured for a cluster that contains six nodes. When one node is faulty, the EC scheme is automatically adjusted from 4+2 to 3+2. In this way, new write I/Os are not interrupted and service continuity is ensured.

For the $N+M$ EC scheme, the value of M remains unchanged, and the value of N is reduced based on the maximum value of N that can be selected. For example, in EC scheme 4+2 (6 nodes), if one node is faulty, the EC scheme is reduced to 3+2. If another node is faulty, the EC scheme is reduced to 2+2. After an EC scheme is reduced, EC reduction is performed only for new data written into the system and the old data remains unchanged. The minimum reduction ratio is $N/2$.

For details about EC reduction, see the *Huawei OceanStor Pacific Series Elastic EC Technical White Paper*.

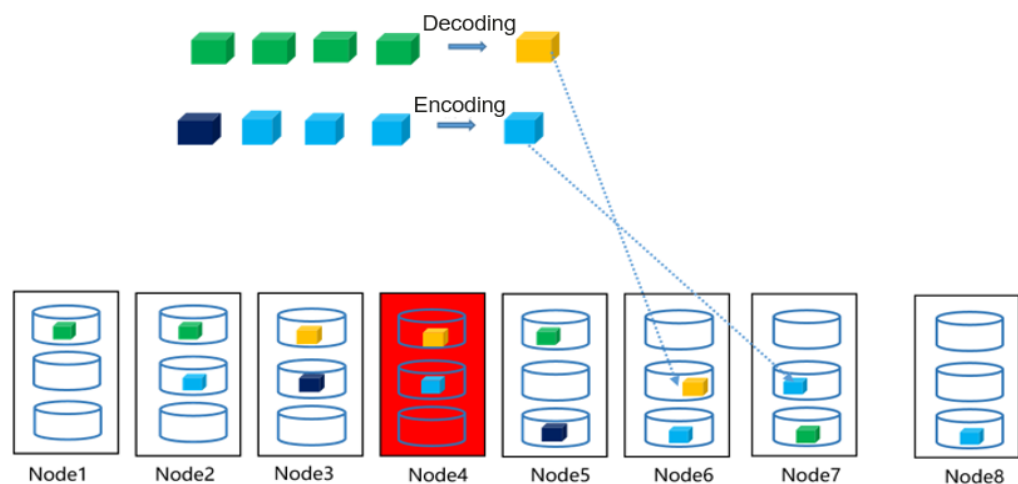
7.3.1.1.4 Fast Data Rebuilding

When a disk or node is offline (for example, a disk is removed or a node is restarted) for a short period of time and is re-connected to the storage system, the system does not rebuild all data on the disk but directly carries out status negotiation (for example, location of the valid data). After the negotiation, data on the disk is directly used. Compared with migrating newly-written data to the recovered disk, this reduces adverse impacts on system performance and improves data reliability. If the disk or node is offline for a long time, data rebuilding is triggered to ensure data reliability and service availability.

Storage pools of Huawei OceanStor Pacific series use the partitioning mechanism. That is, EC stripes are distributed to different disks of different nodes based on specific policies. When detecting that disk or node hardware is faulty (for example, offline for a long time), Huawei OceanStor Pacific series automatically starts data restoration in the background.

OceanStor Pacific series adopts a virtualization technique. Each disk shares a part of its space to form an EC group. Once a disk is faulty, a large number of disks help rebuild the faulty disk's data. In addition, the hot spare space is not located on only one disk but randomly allocated on disks in the storage pool.

Figure 7-5 Data rebuilding of Huawei OceanStor Pacific series



As shown in the preceding figure, if node 4 is faulty, node 6 and node 7 participate in the rebuilding at the same time, improving the rebuilding speed.

OceanStor Pacific series uses Huawei-developed LDEC codes. By maximizing the parity columns of public data codes, the system reads as many public data blocks

as possible during rebuilding. Compared with Reed-Solomon encoding, rebuilding requires 0.75 of the original network bandwidth. Compared with Reed-Solomon encoding, the rebuilding time is shortened by 32.45%.

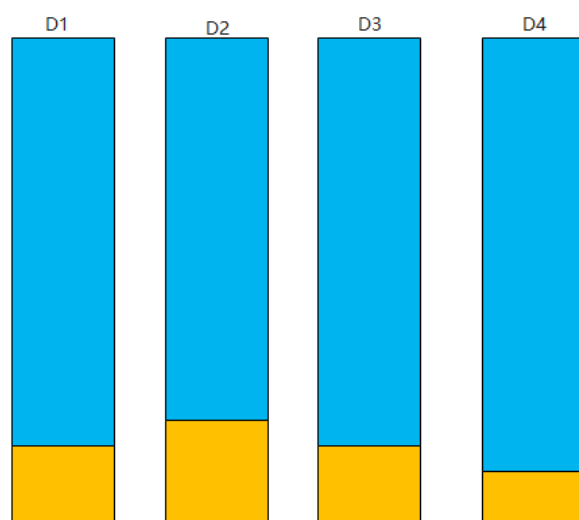
Incremental rebuilding upon a cache disk fault:

When HDDs are used as the main storage and SSDs are used as the cache, if an SSD is faulty and the cache disk is replaced, two data rebuilding mechanisms are available in the industry:

- (1) Full-disk rebuilding. Data on cache disks is unknown. During rebuilding, data written to HDDs needs to be fully restored. A large amount of data needs to be rebuilding. Therefore, the rebuilding time is calculated by day.
- (2) Incremental rebuilding. Data in cache disks is clear. During rebuilding, incremental restoration needs to be performed only on data that has not been written to HDDs in cache disks. The amount of data involved in rebuilding is small, and rebuilding can be completed within a short period of time.

Huawei OceanStor Pacific series innovatively supports incremental data rebuilding upon cache disk failures. This method is based on Plog semantics. It records each piece of Plog data that has been written from SSDs to HDDs. If a cache disk is faulty, Plog metadata can be loaded from HDDs. During rebuilding, incremental restoration needs to be performed only on data that has not been written to HDDs in cache disks. The data that is not flushed to disks in the SSD cache is only about 1% of the HDD space. Compared with full-disk rebuilding, the rebuilding time is reduced by dozens of times, greatly improving the rebuilding efficiency. According to test results, the data rebuilding time upon cache disk failures in Huawei OceanStor Pacific series can be shortened to hours.

Figure 7-6 Incremental data rebuilding



As shown in the preceding figure, D1 to D4 correspond to four data fragment Plogs, respectively. The blue parts indicate data that has been written from SSDs to HDDs, and the yellow parts indicate data that has not been written to HDDs. In addition, the location where the data has been flushed is recorded in the metadata of a fragment Plog. If the cache disk corresponding to data fragment D1 is faulty, after the cache disk is replaced, only the data in the yellow parts (lost) in

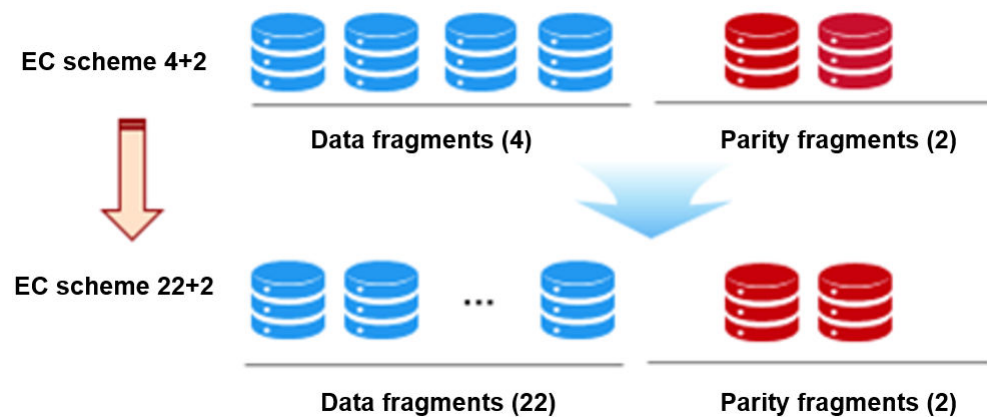
the figure needs to be rebuilt based on the incremental data rebuilding mechanism.

7.3.1.1.5 Dynamic Increase of EC Fragments

EC schemes are closely related to the number of nodes in a storage pool. For example, if a storage pool has three nodes, only EC scheme 4+2:1 can be configured; if a storage pool has four nodes, only EC scheme 6+2:1 can be configured. The EC capacity utilization of a small-scale storage pool is far lower than that of a large-scale storage pool.

After the capacity of a storage pool is expanded, the system supports dynamic increase of EC fragments to improve space utilization. For example, EC scheme 4+2 can be expanded to EC scheme 22+2 after one or more increases of fragments, improving space utilization from 66.6% to 91.67%. In addition, because the number of parity fragments (M) remains unchanged, the reliability does not deteriorate after an increase of fragments.

Figure 7-7 Dynamic increase of fragments



The system provides flexible capacity expansion options, including capacity expansion without fragment increase and capacity expansion with fragment increase. The recommended configuration (optimal EC scheme calculated based on the number of nodes) or specified EC scheme can be used for fragment increase.

After capacity expansion with fragment increase is started, the expanded capacity and the original space where no data is written are immediately converted to the capacity in the new EC scheme. Considering factors such as service pressure and capacity watermarks, the space where data has been written is intelligently converted to the capacity in the new EC scheme by background tasks. Compared with data rebuilding triggered by operations such as node addition and removal, background re-layout triggered by EC fragment increase takes a longer time, but has less impact on user services.

7.3.1.2 Data Integrity Protection

Huawei OceanStor Pacific series resolves silent data corruption problems using mechanisms including I/O-granular real-time end-to-end data integrity check, periodic background data check, and real-time self-healing and correction of corrupted data.

7.3.1.2.1 I/O Path Data Integrity Protection

During data transmission within a storage system, data passes through multiple components over various channels and undergoes complex software processing. Any problem that occurs during the transmission may cause data errors. The scenario where a problem is not immediately detected but found in subsequent data access is called silent data corruption. As silent data corruption cannot be detected in real time, corrupted data is hard to or even impossible to recover.

Possible causes of silent data corruption are as follows:

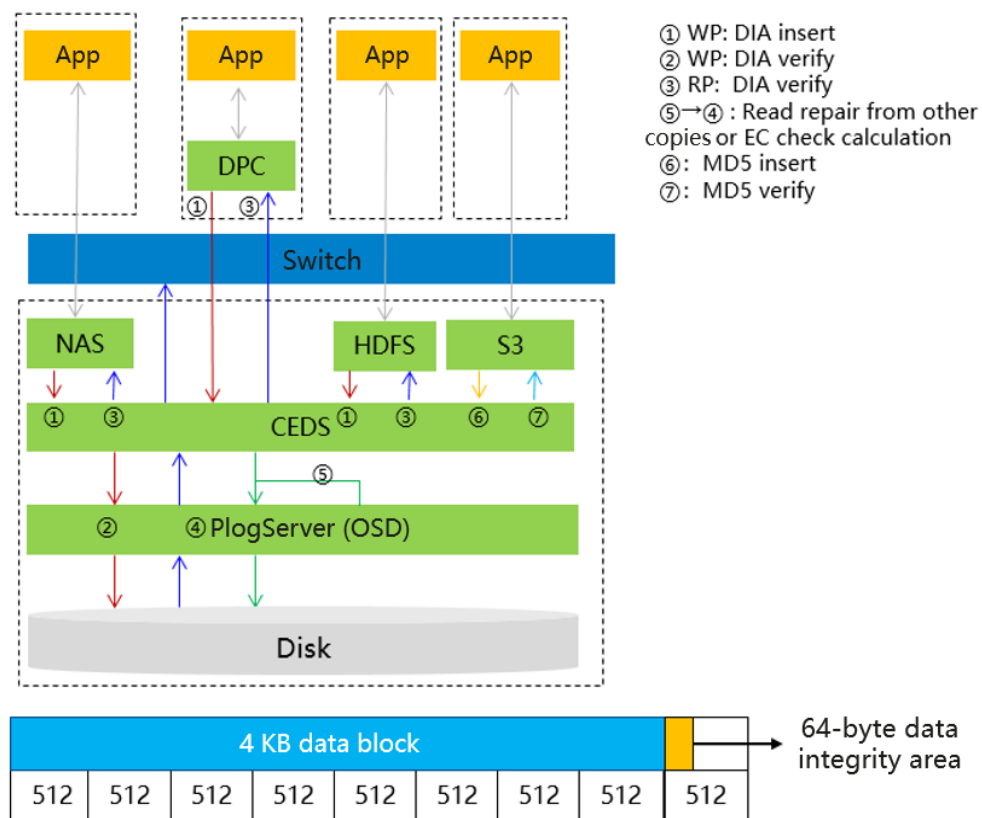
- Hardware faults: faults in memory modules, CPUs, disks, and FC or SAS links
- Firmware faults: faults in HBAs and disks
- Software bugs: bugs of software, operating systems, and applications

Data errors caused by hardware faults can be detected using error checking and correction (ECC) or cyclic redundancy check (CRC). Silent data corruption is more possible to be caused by firmware faults or software bugs.

Huawei OceanStor Pacific series provides an I/O-level end-to-end data integrity protection solution, which can detect various silent data corruption scenarios, such as bit changes and read/write offset. When detecting silent data corruption, the system performs correction and self-healing on data in real time using data redundancy to prevent data corruption from spreading.

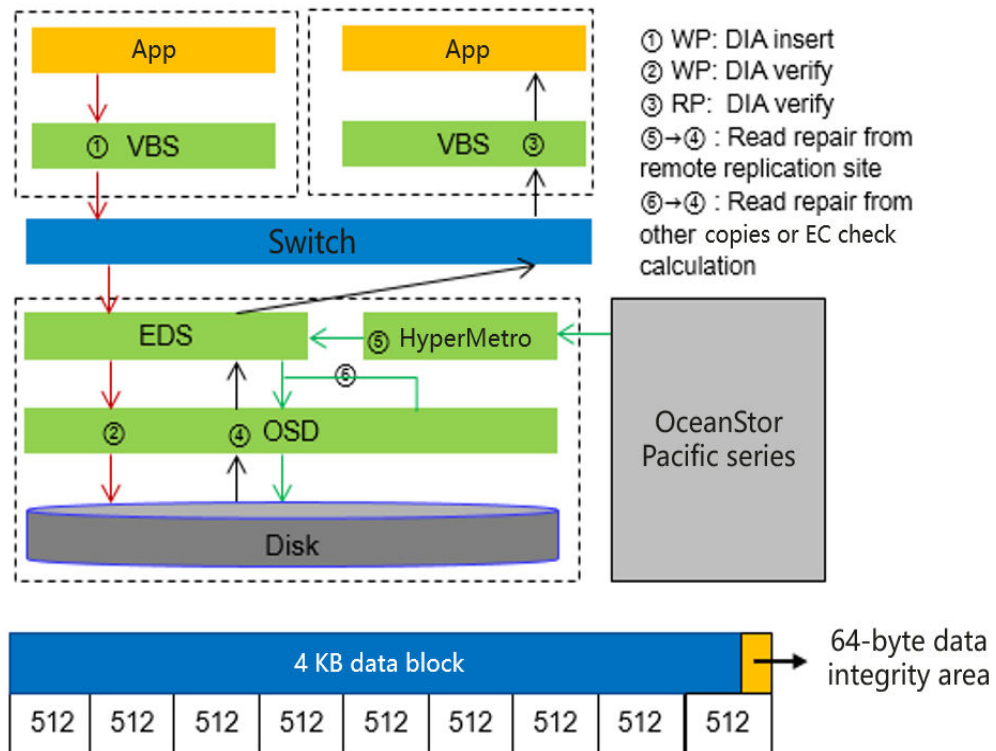
The following figure shows the error detection location of key silent data in the I/O path and the disk layout. The file service is used as an example.

Figure 7-8 Error detection location of key silent data in the I/O path and disk layout of the file service



The following figure shows an example of the structured service.

Figure 7-9 Error detection location of key silent data in the I/O path and disk layout of the structured service



The size of a disk sector can be 512 or 520 bytes.

Data integrity area (DIA): data integrity check area

Block, NAS, HDFS, and S3 services support end-to-end check, and the object service supports object-level MD5 check.

7.3.1.2.2 Periodic Check of Persistent Data

Silent data corruption may occur on an HDD or SSD due to component aging, electromagnetic interference, signal interference, or process defects. Periodic data check can identify and handle risks in advance, which prevents data loss caused by accumulation of silent data corruption.

Huawei OceanStor Pacific series uses a self-adaptive background periodic check mechanism to prevent data errors. The process is as follows:

1. The system delivers a read I/O to read data from an HDD or SSD.
2. The system calculates the DIA of the data blocks to be read (DIA') and compares it with that of the read data (DIA).
3. If DIA and DIA' are different, the re-read verification is triggered to determine whether the data inconsistency is temporarily caused by links. If they are still different, the system inserts the data into the data correction and self-healing queue to perform a background self-healing write repair.
4. The rate of background verification tasks can be manually adjusted. The priority of the triggered I/Os is lower than that of host services. When a host

service is running, it is processed preferentially. This ensures that host services are not interrupted.

7.3.1.2.3 Correction and Self-Healing of Corrupted Data

The storage system supports correction and self-healing of corrupted data. No matter silent data corruption is detected in host I/Os or background periodic I/Os, automatic user-unaware error correction and self-healing for corrupted data will be triggered. Error correction and self-healing can rely on redundant data in storage media on a storage node of the local system.

7.3.1.3 Restoration of Mistakenly Deleted Data

Unstructured services of OceanStor Pacific series provide the file system/dtree-based recycle bin. When a user deletes files, the files are moved to the recycle bin directory in the background so that the user can retrieve the data. The recycle bin function is valid for all files and objects in unstructured namespaces.

For details about the recycle bin, see the *Huawei OceanStor Pacific Series Recycle Bin Technical White Paper*.

7.3.2 Sub-health Management

Sub-health indicates that the hardware can run properly but the performance severely deteriorates. Hardware sub-health can be caused by various reasons, including but not limited to firmware bugs, hardware design defects, temperature, environment (such as vibration), and incorrect configurations.

A variety of hardware devices may be in sub-healthy state, including but not limited to disks (SSDs and HDDs), networks, CPUs, and memory modules. Once the hardware becomes sub-healthy and no effective monitoring and fault tolerance measures are taken in the storage system, the latency of the storage system responding to hosts may increase and IOPS/BPS may be reduced. Furthermore, the system cannot respond to hosts and services on the hosts are interrupted. OceanStor Pacific series monitors disks, networks, and services (including CPUs and memory modules) in sub-healthy state, implements intelligent diagnosis and automatic isolation. In this way, a single sub-healthy component does not affect the storage system performance, and the hosts can experience the consistent performance.

7.3.2.1 Disk Sub-health Management

Huawei OceanStor Pacific series comprehensively monitors main storage disks, cache disks, system disks, and metadata disks. The monitored contents include but are not limited to the following:

- SMART information (for example, whether the number of disk sector remaps exceeds the threshold and whether the read error rate exceeds the threshold)
- I/O latency (including single I/O latency and average I/O latency)
- I/O errors (including the number of silent data errors)

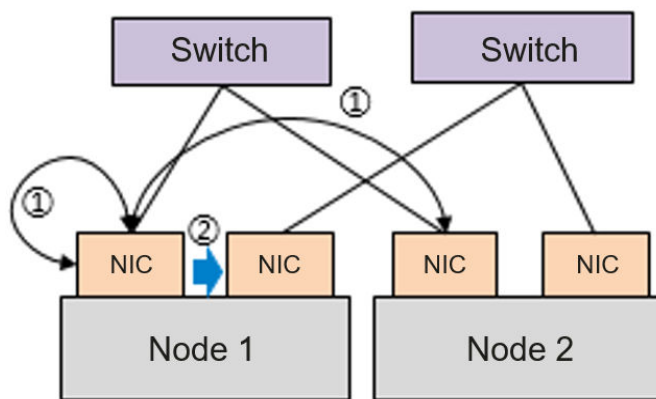
7.3.2.2 Network Sub-health Management

Cluster networks become sub-healthy when network performance deteriorates due to decreased NIC speed, increased packet loss rate or packet error rate, or

incorrect negotiation rate. The system detects the changes in network resource status to locate the nodes affected by network sub-health, and performs an active/standby switchover for bond ports or isolates the nodes. The basic working principles are as follows:

- Multi-level detection: The local network of a node quickly detects exceptions such as intermittent disconnections, packet errors, and low negotiated rates. In addition, nodes are intelligently selected to send detection packets in an adaptive manner to identify link latency exceptions and packet loss.
- Smart diagnosis: Smart diagnosis is performed on network ports, NICs, and links based on networking models and error messages.
- Level-by-level isolation and warning: Network ports, links, and nodes are isolated based on the diagnosis results and alarms are reported.

Figure 7-10 Local network port switchover triggered by network sub-health



The following describes how a local network port switchover is triggered by network sub-health.

1. Node 1 continuously sends detection packets to other storage nodes in the cluster to check whether packet loss occurs or the latency increases.
2. Node 1 detects that the detection packets sent from the current network port to multiple target storage nodes are abnormal, and performs a network port switchover. After the switchover, the network service recovers.

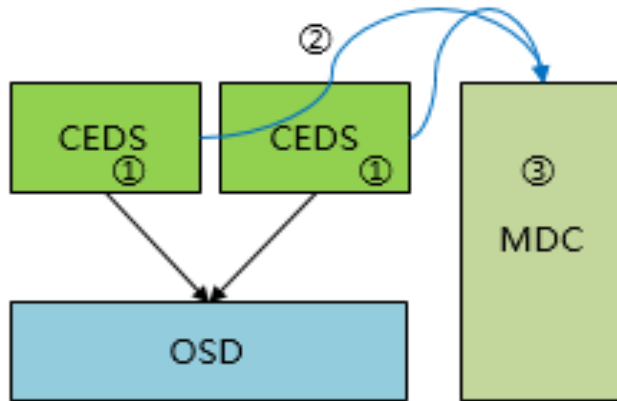
7.3.2.3 Service Sub-health Management

It is common that software or hardware faults occur during the running of nodes in a scale-out cluster. These faults render the nodes sub-healthy, causing problems such as access rate decrease due to repeated memory error correction and CPU underclocking. In this case, the system service latency increases. The system locates the sub-healthy nodes by collecting latency statistics and isolates the nodes or resources. The basic working principles are as follows:

- Cross-process/service detection: The system collects statistics about the I/O latency of cross-process/service access. If the I/O access latency exceeds the threshold, a comprehensive diagnosis is reported.
- Smart diagnosis: The system diagnoses processes or services with abnormal latency using the majority voting or clustering algorithm based on the reported abnormal latency of each process or service.

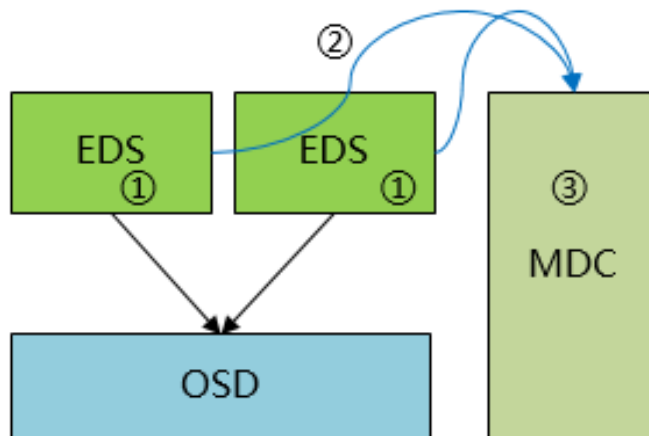
- Isolation and warning: Abnormal processes or services are reported to the control node for isolation (services are distributed to other normal processes in the cluster) and alarms are reported.

Figure 7-11 Sub-health detection and isolation of unstructured services



1. A CEDS process detects the latency of an OSD process and checks whether the latency continuously increases.
2. A CEDS process detects that the latency of an OSD service is abnormal and reports the exception to MDC.
3. MDC determines whether most CEDS processes that access the OSD service report the abnormal latency of the OSD service. If yes, the OSD process is isolated.

Figure 7-12 Sub-health detection and isolation of the structured service



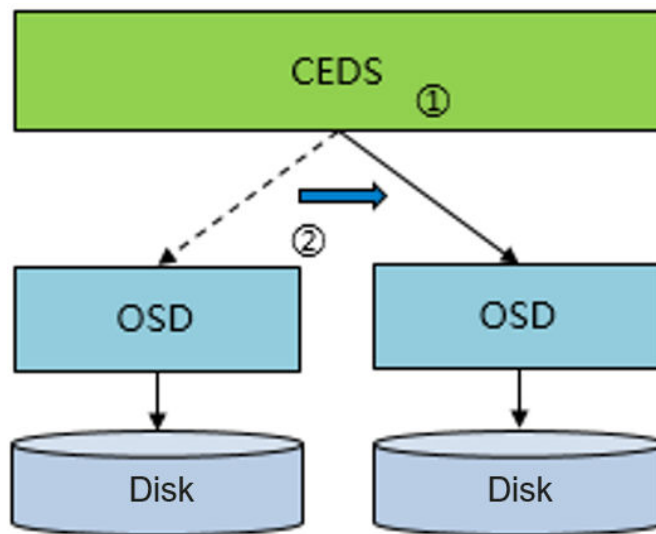
1. An EDS process detects the latency of an OSD process and checks whether the latency continuously increases.
2. An EDS process detects that the latency of an OSD service is abnormal and reports the exception to MDC.
3. MDC determines whether most EDS processes that access the OSD service report the abnormal latency of the OSD service. If yes, the OSD process is isolated.

7.3.2.4 Fast-Fail Mechanism

Huawei OceanStor Pacific series provides dedicated fast retry of path switching, that is, the Fast-Fail feature, to ensure that the I/O latency of a single sub-healthy node is controllable. The following mechanisms are used to implement the feature:

- Average I/O latency detection: Checks whether the average I/O latency exceeds the threshold and whether a response is returned for the I/O. If no response is returned, Fast-Fail is triggered.
- Fast-Fail: For read I/Os, copies on other nodes are read or degraded read is performed after path switching. For write I/Os, space is reallocated on other disks to store data.

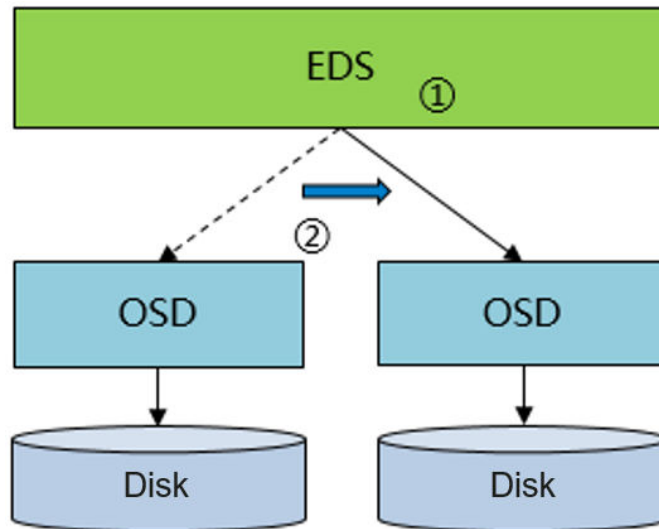
Figure 7-13 Fast-Fail of unstructured services



The following is an example of how to trigger Fast-Fail for a slow disk of unstructured services:

1. A CEDS process collects latency statistics of I/Os delivered to disks and identifies the disks whose access latency is longer than that of normal disks using the clustering algorithm.
2. The CEDS process marks the abnormal disk and forwards the read and write I/Os to other healthy disks to recover the read and write latency.

Figure 7-14 Fast-Fail of the structured service



The following is an example of how to trigger Fast-Fail for a slow disk of the structured service:

1. An EDS process collects latency statistics of I/Os delivered to disks and identifies the disks whose access latency is longer than that of normal disks using the clustering algorithm.
2. The EDS process marks the abnormal disk and forwards the read and write I/Os to other healthy disks to recover the read and write latency.

7.4 Solution-Level Reliability Design

7.4.1 Local Data Protection

Local data protection is used to implement data backup in a cluster at a site to ensure service continuity.

Local data protection of Huawei OceanStor Pacific series is implemented using storage snapshots. A snapshot is an available copy of the specified data set. The copy contains the static image of the source data at the point in time when the source data is copied. A snapshot serves as a data backup and is accessible to hosts. The snapshot feature of Huawei OceanStor Pacific series has the following advantages:

- Snapshots and their source data share the same storage space, without the need to plan exclusive storage space for snapshots.
- You can use a snapshot to create multiple data copies that are independent of each other and accessible to application servers for various purposes, such as data testing, archiving, and analysis. In this way, source data is protected, while backup data serves new purposes, meeting enterprises' various data needs.

Creating a snapshot is equivalent to creating a protected read-only version. When you write or delete data, the existing data space is protected by the snapshot and

thus remains unchanged. New data will be written to new storage space in ROW mode.

For details about the snapshot technology, see the *Huawei OceanStor Pacific Series HyperSnap Technical White Paper* for unstructured services and *Huawei OceanStor Pacific Series HyperSnap Technical White Paper (Block)* for the structured service.

7.4.2 Site-Level Data Protection

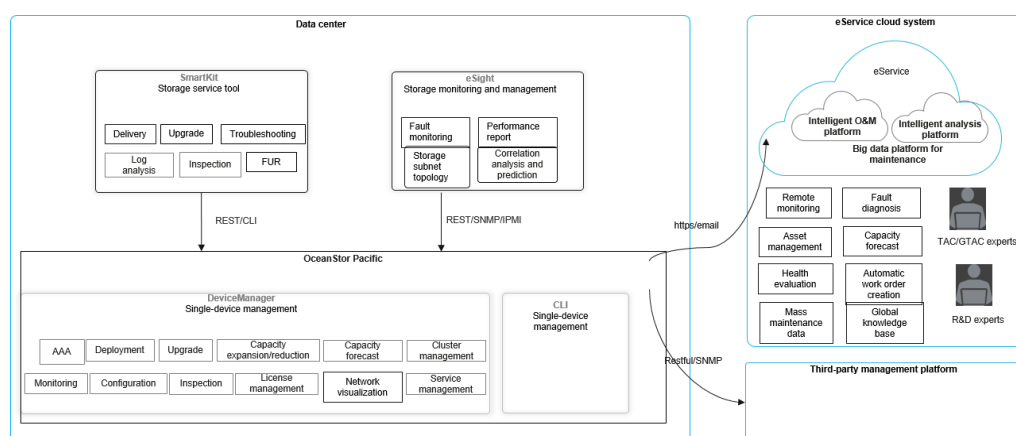
Site-level data protection uses the data replication technology between sites to ensure service continuity. The asynchronous replication technology is used to construct DR solutions for OceanStor Pacific series unstructured services. OceanStor Pacific series structured service supports DR solutions based on asynchronous/synchronous replication and active-active technologies to ensure service continuity.

OceanStor Pacific series asynchronous replication periodically synchronizes data between primary and secondary storage systems for system DR. This minimizes service performance deterioration caused by the latency of long-distance data transmission.

The HyperMetro solution provided by the OceanStor Pacific series structured service enables active-active DR between two OceanStor Pacific series block storage clusters. Based on the volumes of the two OceanStor Pacific series block storage clusters, a virtual volume is provided and the real-time data on the volume can be synchronized between the two clusters. In addition, the two clusters can simultaneously respond to the read and write I/O requests from application servers, allowing concurrent access. If one data center fails, the other one quickly takes over services without data loss and service interruption.

8 System Serviceability Design

Figure 8-1 OceanStor Pacific series management diagram



OceanStor Pacific series supports four types of services: file, object, HDFS, and block. Unstructured services can be deployed in the same storage pool. OceanStor Pacific series uses DeviceManager, a graphical user interface (GUI), to manage clusters. DeviceManager Client provides an initialization wizard to guide users to install management nodes, add storage nodes, configure storage networks, install storage nodes, and create control clusters. In addition, it provides various APIs, such as RESTful APIs and SNMP, and supports OpenStack SMI-S, facilitating integration with third-party management platforms.

OceanStor Pacific series also supports O&M management such as Global center management on Huawei eSight, cloud-based management on Huawei eService, and intelligent inspection on Huawei SmartKit.

8.1 System Management Tools

8.2 Resource Management for Storage Services

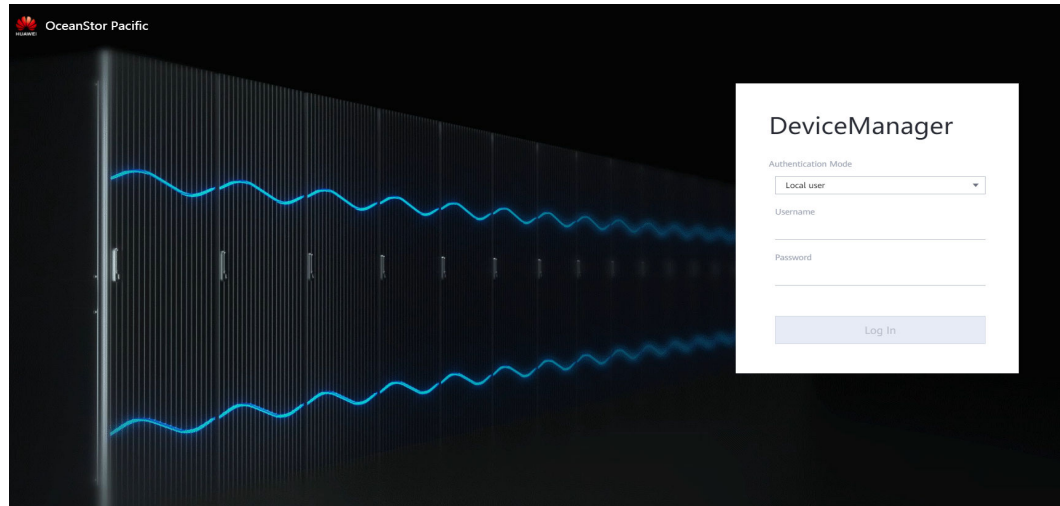
8.3 System O&M Management

8.1 System Management Tools

8.1.1 DeviceManager

DeviceManager is a built-in graphical user interface (GUI) management tool of Huawei OceanStor Pacific series. Users can access **<https://Storage management IP address:8088/>** using a common browser. You can use DeviceManager to complete almost all required configuration and management operations. The following figure shows the DeviceManager login page.

Figure 8-2 DeviceManager login page



Management users of OceanStor Pacific series DeviceManager can be classified into system administrators, system operators, and system viewers by role. The functions of DeviceManager can be divided into the following categories:

- **Resource addition**
Resource addition refers to adding nodes to OceanStor Pacific series. Nodes can be added one by one or in a batch. Before performing batch import, download a batch import template from DeviceManager. Before adding a node, you need to install the operating system and configure IP addresses, and then enter the management IP address of the node on DeviceManager. After adding a node, you can configure the storage network and install storage nodes for the node on DeviceManager.
DeviceManager provides a wizard for adding new nodes. The wizard includes the following steps: adding storage nodes, configure storage networks, and installing software.
- **Resource management**
Resource management includes storage pool management and unique resource management for different services. Storage pool management allows you to create storage pools, view statistics about storage pools, view the disk topology of storage pools, expand or reduce the capacity of storage pools, modify storage pools, add storage pools to QoS policies, and delete storage pools.
- **System management**
System management includes the following:

- Basic cluster information monitoring: allows you to view basic cluster information, including the cluster name, status, node information, and node process information.
- Performance monitoring: allows you to view the CPU usage, memory usage, bandwidth, IOPS, latency, disk usage, and storage pool usage.
- Alarm management: allows you to view alarms, configure alarm notification by email, Syslog, or SNMP, and dump, clear, mask, and export alarms.
- User management: allows system administrators to create new administrators and grant them management permissions for managing the system or resources. You can create, query, modify, delete, unlock, and bring offline users and configure password policies to enhance system security on DeviceManager.
- License management: allows you to view activated licenses and import new ones.
- Hardware management: allows you to centrally manage all nodes, disks, and network ports in the system.
 - Allows you to view names, management IP addresses, storage IP addresses, health status, running status, software installation status, cabinets, and slots of nodes and set nodes to the maintenance mode to facilitate fault rectification.
 - Allows you to view names, status, slots, serial numbers, and types of disks.
 - Allows you to view names, status, MAC addresses, protocol types, and IP addresses of network ports.
- Network topology management: allows you to query the topology information about the management network, storage network, and replication network.

Task center management: allows you to query the progress and status of background tasks.

8.1.2 CLI

The Command Line Interface (CLI) allows administrators and other system users to manage and maintain the storage system. The CLI is based on the SSH protocol and supports key-based SSH access. For details, see [OceanStor Pacific Series 8.1.3 Command Reference](#).

8.1.3 RESTful API

OceanStor Pacific series uses open REST APIs to integrate with non-OpenStack cloud management platforms. Upper-layer cloud management platforms can use REST APIs to monitor and manage storage system resources, such as storage clusters, storage pools, and volumes.

All the APIs comply with HTTP 1.1 (RFC 2616) and use the REST style. Each request contains the POST and GET methods of HTTP. Each response to a request contains the standard HTTP status code. Major management APIs are as follows:

- Storage pool management APIs: monitor storage pool information, including the capacity (total capacity, allocated capacity, and used capacity), status, security level, and redundancy policy.
- Volume management APIs: manage storage volumes, including creating volumes, mapping volumes to hosts, mounting volumes, deleting volumes, expanding volume capacity, unmounting volumes, and querying volumes.
- Snapshot management APIs: manage storage snapshots, including creating snapshots, deleting snapshots, querying snapshots, creating volumes using snapshots, and creating consistency snapshots for multiple volumes.
- System user management APIs: manage system management users, including creating accounts, deleting accounts, and changing or resetting account passwords.
- Performance monitoring APIs: monitor performance of storage pools, hosts, and disks, such as storage space utilization, read/write IOPS, read/write bandwidth, and latency.
- Installation and deployment management APIs: enable automatic deployment, including adding hosts, querying hosts, and installing software.
- Operation log query APIs: query operation logs by operation name, operation result, or operation time.

8.1.4 SNMP

OceanStor Pacific series supports standard SNMP v1/v2/v3 management protocol interfaces. SNMP users can query and set system information and receive system alarm information.

8.1.5 SmartKit

Huawei SmartKit inspection tool contains knowledge bases of different product fields, making operations easier for maintenance personnel. Concurrent multi-gateway or multi-process inspection greatly improves the inspection efficiency. It has the following highlights:

- Enables the lifecycle management team (LMT) to continuously update inspection instances based on maintenance experience and precaution notices, and release corresponding scripts in a timely manner. SmartKit can then be automatically updated.
- Automatically outputs analysis reports and statistics reports based on recorded inspection packets.
- Automatically obtains an NE list from the network management system (NMS) to perform remote and batch inspection with high efficiency.

8.1.6 eSight

Huawei eSight is used to centrally manage storage devices of different manufacturers in an enterprise data center. It provides diversified management functions such as a global topology view, capacity analysis, performance analysis, health analysis, fault diagnosis, and end-to-end visual management, greatly improving the storage network O&M efficiency. It has the following highlights:

- Centrally manages 100+ storage devices from multiple vendors, simplifies storage management, and provides a unified device management view to centrally display storage hardware and logical resources.
- Monitors devices in real time, displays system information indicators in graphics, and automatically reports alarms, requiring no administrator attendance.
- Outputs intelligent analysis reports to monitor key services, laying a foundation for proper space planning.

8.1.7 eService

Huawei eService is an intelligent cloud O&M platform. It uses big data analysis and intelligent technologies to provide services such as automatic fault reporting, capacity prediction, performance prediction, disk risk prediction, and fault handling progress tracking for data infrastructure such as Huawei storage devices and servers. It provides proactive and predictive O&M of data infrastructure anytime and anywhere, simplifying O&M and improving O&M efficiency. Huawei eService cloud big data intelligent analysis system provides the following functions:

- Intelligent storage management services, such as fault prediction (risky disk prediction), capacity management, and performance anomaly analysis
- Proactive O&M services, including all-round proactive protection and fast troubleshooting, reducing accident risks and O&M costs

8.2 Resource Management for Storage Services

8.2.1 Resource Management for the File Service

- File system management
The file service of OceanStor Pacific series supports multiple namespaces, that is, multiple file systems. A single storage cluster supports a maximum of 100,000 file systems. A storage administrator can create, delete, and query file systems on DeviceManager. Each file system is an independent namespace. User data in different namespaces is isolated.
- Dtree management
A directory tree (dtree) is a special directory used to configure value-added features, such as HyperSnap, SmartQuota, and SmartTier. Dtrees must be created or deleted on DeviceManager, in the CLI, or using RESTful interfaces. Dtrees can be created in any directory including the root directory of a file system. However, dtrees cannot be nested.
- Service IP address management
OceanStor Pacific series is a fully symmetric file system. Each storage node can provide storage services. Therefore, IP addresses must be configured for each node. In a scale-out storage system, a large number of IP addresses need to be configured. To facilitate the use of the storage system, OceanStor Pacific series provides the domain name service through which users can access the storage system. When domain names are resolved and service IP addresses are returned, the system implements load balancing based on the configured load balancing policy (round-robin, CPU usage, or node connections).

- DPC management
After deploying DPCs, you need to create a DPC cluster on DeviceManager and add the compute nodes where DPCs have been installed to the DPC cluster. Only the nodes that have been added to the DPC cluster can access storage services.

8.2.2 Resource Management for the Object Service

- Authentication management
OceanStor Pacific series supports built-in POE authentication, as well as external AD domain and LDAP domain authentication.
- Unified namespace management
 - Configuring the LS
Configure public and private addresses for the global LS, addresses for the regional LS, and a global domain name.
 - Managing regions
Adding regions: Add non-default regions to implement global unified management.
Adding clusters: Add clusters to a local region to centrally manage clusters.
Configuring the domain name resolution service: Configure the domain name resolution service for the object service.
- Security management
 - Security access policy
 - Enabling the IP address AK policy
If, within a statistical period, the number of access failures of an AK of an IP address is greater than or equal to the threshold for access failures, and the percentage of the number of access failures to the total number of access attempts is greater than or equal to the threshold for the access failure rate, services for the AK and IP address are denied.
 - Enabling the AK policy
If, within a statistical period, the number of access failures of an AK is greater than or equal to the threshold for access failures, and the percentage of the number of access failures to the total number of access attempts is greater than or equal to the threshold for the access failure rate, services for the AK are denied.
 - Enabling the IP address access policy
If, within a statistical period, the number of access failures of an IP address is greater than or equal to the threshold for access failures, and the percentage of the number of access failures to the total number of access attempts is greater than or equal to the threshold for the access failure rate, services for the IP address are denied.
 - TLS policy
You can configure TLS policies as required.

- Time verification
Access configurations include domain name and time verification for the object service.
- Enabling time verification
When time verification is enabled, access requests will be rejected if the time difference between clients and servers exceeds 15 minutes.
- Others
 - Configure a namespace for object service responses.
 - Redirect to a URL if access to a hosted static website fails.

8.2.3 Resource Management for the HDFS Service

- Account management
The HDFS service implements system resource isolation based on accounts and uses accounts as the basic management unit. The HDFS service adopts AKs and SKs to authenticate user identities, and employs Kerberos authentication and LDAP user management during data processing.
- Namespace management
You can configure namespaces and set quotas for them, as well as view the namespace list and quota usage.
- LDAP user management
You can create local users and local user groups to manage namespaces. The HDFS service uses LDAP authentication to share namespaces.
- Load balancing
The HDFS service supports zone-based management of clusters to effectively control data flows and improve service performance and reliability.

8.2.4 Resource Management for the Block Service

- Storage pool management
You can view the statistics and disk topology of the selected storage pool, expand or reduce the capacity of the selected storage pool, and delete a storage pool. You can also create a storage pool.
- Block storage client management
You can create and delete block storage clients. You can also view the mounting information of a block storage client as well as the CPU and memory monitoring statistics to mount or unmount volumes on the client.
- Volume management
You can create and delete volumes. When creating a volume, you need to specify a resource pool to which the volume belongs and set the volume name and size. If the created volume is used based on the SCSI protocol, the volume needs to be mounted to a host. If the volume is used based on the iSCSI protocol, you need to map the volume to a host or host group using the iSCSI protocol. On the DeviceManager page for mapping volumes using iSCSI, you can create hosts or host groups, configure initiators, configure CHAP authentication, and map volumes to or unmap volumes from hosts or host groups.

 NOTE

By default, the iSCSI service is disabled. To use the iSCSI service, enable it and add the IP addresses and ports used for iSCSI bonding.

- **QoS policy management**
You can create and delete QoS policies, and check the QoS policy information on multiple pages.
- **Snapshot management**
You can create linked clone volumes, set QoS policies, delete snapshots, and view the snapshot list on multiple pages. The list information includes the snapshot name, capacity, owning storage pool, and creation time.

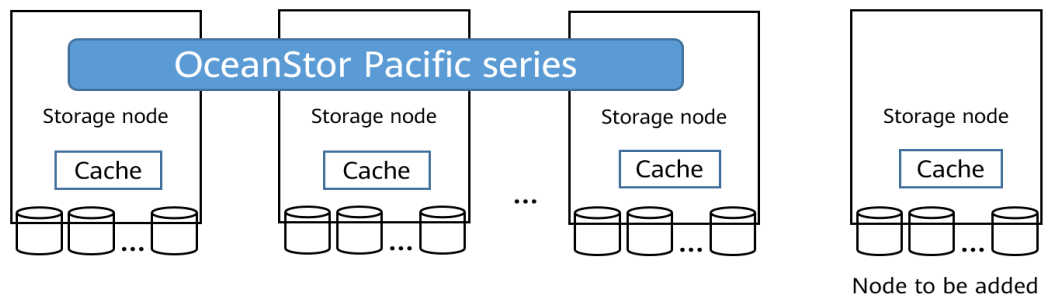
8.3 System O&M Management

8.3.1 Node Addition and Deletion

OceanStor Pacific series adopts a scale-out architecture with superb scalability and supports ultra-large capacity storage.

- After new nodes are added, the system implements fast load balancing, avoiding migration of a large amount of data.
- There is no independent storage node. Storage access requests, storage bandwidth, and cache resources are evenly distributed on each node. The system IOPS, throughput, and cache linearly increase as nodes expand.

Figure 8-3 Capacity expansion of OceanStor Pacific series



In addition, OceanStor Pacific series supports node deletion. The system automatically migrates valid data on the node to be deleted to meet users' space management requirements.

8.3.2 System Upgrade

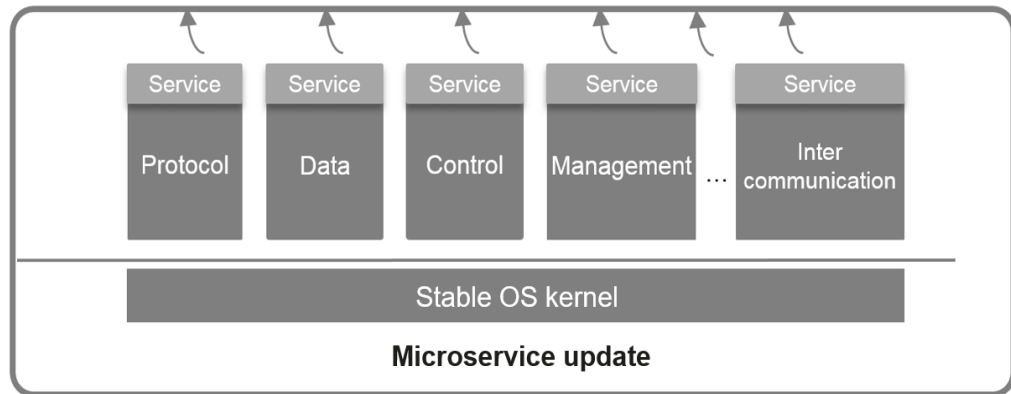
OceanStor Pacific series adopts a cloud-oriented design and scale-out architecture to provide long-term reliable storage services. Microservices are automatically upgraded to ensure continuous updates of storage services and provide stable storage services.

The modular design of OceanStor Pacific series enables storage services to be upgraded or updated by upgrading or updating microservices.

- Storage service upgrade: Old microservices are deactivated and new microservices are activated to upgrade storage services, for example, to optimize performance and enhance reliability.
- Storage service update: New microservices are loaded to provide new storage service capabilities, for example, to introduce intelligent storage tiering.

Microservice can be activated within five seconds to upgrade or update storage services.

Figure 8-4 OceanStor Pacific series microservice upgrade



The scale-out architecture of OceanStor Pacific series offers high scalability to meet requirements for deploying large-scale clusters. During a long period of running, the storage system may require upgrades due to various reasons, such as introducing new features and upgrading kernels of underlying operating systems. OceanStor Pacific series supports system rolling update. Nodes and storage pools can be rolling updated in batches. This reduces upgrade risks and impacts on services, and improves upgrade efficiency.

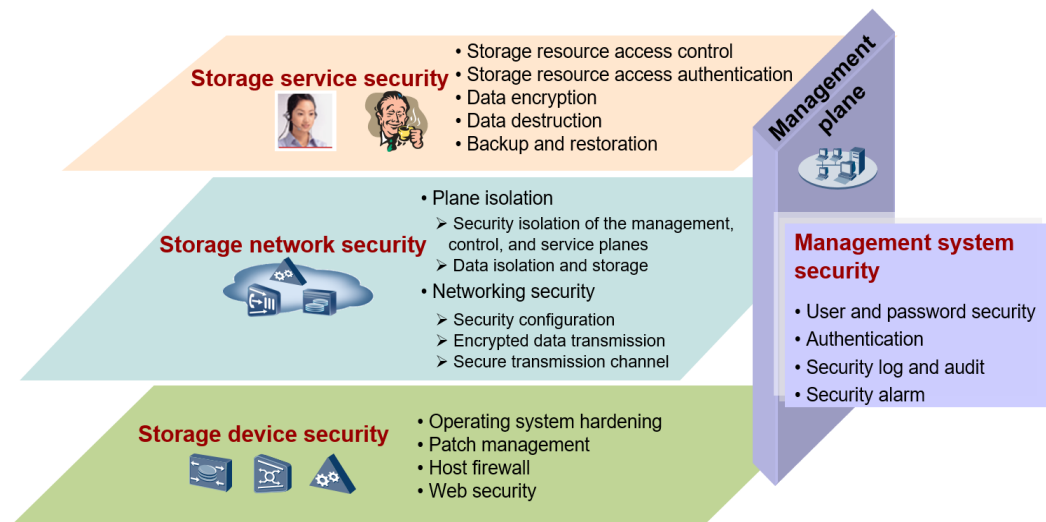
System rolling update can be performed by only one click, which is simple and efficient. It takes only five minutes to upgrade one node in a storage pool. This ensures smooth and stable service performance.

9 System Security Design

As the Internet and cloud technologies develop, IT systems face increasing security threats, including conventional and emerging security threats. In terms of network security, security threats can be classified into those from external networks (such as IP attacks, software vulnerabilities, viruses, Trojan horses, SQL injection attacks, and phishing attacks) and those from internal networks (such as ARP spoofing, malicious plug-ins, unauthorized Internet access, mobile device access, and lack of application monitoring).

Besides the preceding threats, storage systems are subject to data leakage, data damage, and temporary or permanent loss of accessibility and availability. Technical control measures are required to monitor data integrity, confidentiality, and availability, to prevent unauthorized use of storage resources and data.

Figure 9-1 Security architecture of OceanStor Pacific series



The security architecture of OceanStor Pacific series is described as follows:

- Storage device security: operating system hardening, patch management, and web security
- Storage network security: plane isolation and channel security hardening

- Storage service security: storage resource access control, authentication, data encryption, and log audit
- Storage management security: user security hardening, password security hardening, authentication, as well as log and alarm management

[9.1 Storage Device Security](#)

[9.2 Storage Network Security](#)

[9.3 Storage Service Security](#)

[9.4 Storage Management Security](#)

9.1 Storage Device Security

9.1.1 Operating System Hardening

The storage system supports both general and dedicated operating systems. General operating systems are hardened using common security hardening policies. Dedicated operating systems are hardened using the following policies:

- Service minimization
Risky background processes and services are disabled to efficiently prevent operating systems from intrusions and reduce risks.
- File and directory permission control
File permissions are minimized in accordance with security hardening standards and application requirements in the industry. This prevents unauthorized file access due to improper permission settings. For example, common users cannot access files available only to administrators.
- Account security hardening
Unnecessary accounts are deleted to prevent intruders from using these accounts. Different permissions are set for different accounts to prevent unauthorized access.
- Password security hardening
Password complexity check, password validity check, and login retry mechanisms are enabled, preventing brute force password cracking.
- Logging and auditing
Run logs of services and processes and operation logs are recorded for auditing.

9.1.2 Secure Startup

Arm-based storage nodes and EulerOS support the secure startup function to ensure that the devices are not tampered with during the startup. During device startup, the hardware-protected and tamper-proof root of trust (RoT) is executed first. As the first level of the trust chain, RoT verifies the digital signature for the next-level startup code, implementing digital signature verification from the underlying trusted hardware to the OS.

9.1.3 Patch Management

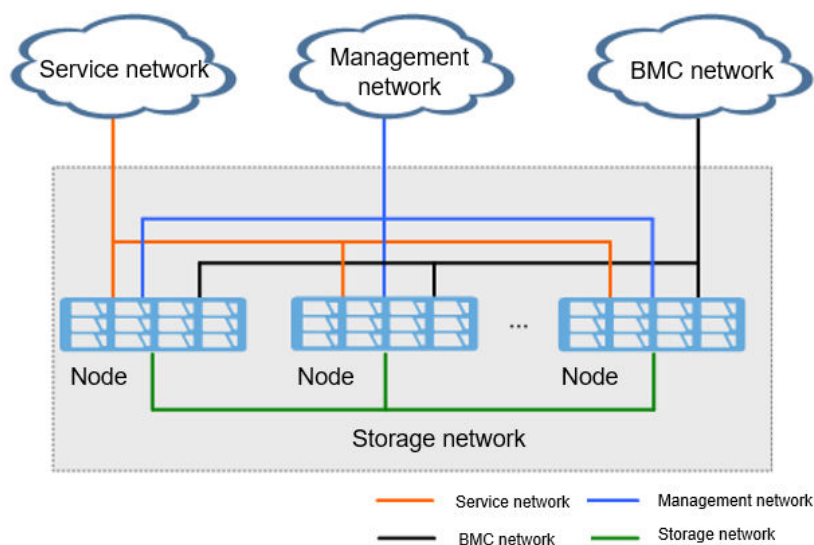
Software design defects result in system vulnerabilities. System security patches must be installed periodically to fix these vulnerabilities and protect the system against attacks by viruses, worms, and hackers. To safeguard system security, Huawei periodically provides on-demand security patches for users as security patches for dedicated operating systems and open-source software are released.

9.2 Storage Network Security

9.2.1 Plane Isolation

OceanStor Pacific series adopts a secure physical network architecture that can be divided into a management network, BMC network, service network, and storage network based on service types. VLANs can be configured to logically isolate these networks, or independent network ports and switches can be used to physically isolate the networks, ensuring secure system running.

Figure 9-2 OceanStor Pacific series cluster networking



9.2.2 Channel Security

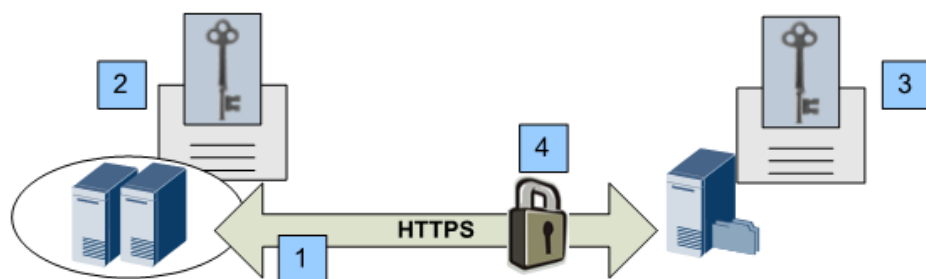
- Secure remote system management

Secure transmission protocols, such as SSH and HTTPS, are used for remote system management, as shown in the following table and figure.

Table 9-1 Secure transmission protocols

Application	Protocol	Port
SSH server (CLI for system management)	SSHv2	22
DeviceManager (GUI for system management)	TLSv1.2	8088 or 28443

Figure 9-3 Secure remote system management



- **Secure remote data transmission**
OceanStor Pacific series supports remote data replication and active-active storage of structured data. Data may be transmitted across untrusted networks. To ensure security of data transmission across untrusted networks, OceanStor Pacific series allows users to configure IPsec devices or VPNs.
- **Secure object data transmission**
The object service provides APIs compatible with Amazon S3. You can upload and download data securely by using terminal tools provided by Huawei or third parties. During data transmission, TLSv1.2 is used by default. TLSv1.1 and TLSv1.0 are also supported.

9.3 Storage Service Security

OceanStor Pacific series storage services provide access control, data encryption, and service access log audit to ensure the security of storage services.

9.3.1 Access Control

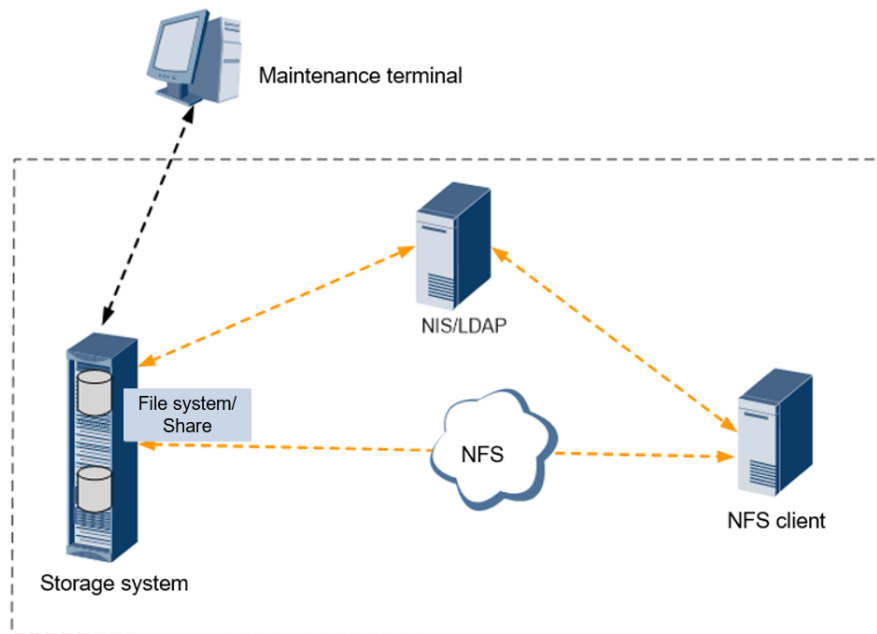
9.3.1.1 File Service Access Control

File service access control is implemented using ACLs. An ACL defines access permissions that users have to a particular system resource. Only users who have the permissions can access the resource.

Currently, NFS and SMB are supported. NFS file access supports local authentication, NIS domain authentication, and LDAP domain authentication.

Hosts and users that can access the system need to be locally configured to implement local authentication. NIS and LDAP domain authentication authenticates IP addresses that access the system using a domain server. SMB file access supports AD domain authentication.

Figure 9-4 NFS domain authentication



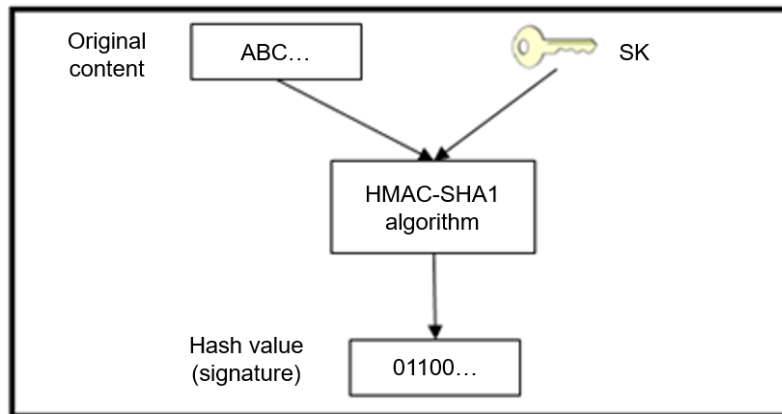
9.3.1.2 Object Service Access Control

The object service provides flexible and secure data access control. You can set different access control policies for buckets and objects as required. The access permissions include READ and WRITE. You can also grant other users the permissions to view and set access control policies for your buckets and objects. Such permissions include READ_ACP (viewing current access control policies of buckets and objects), WRITE_ACP (setting access control policies of buckets and objects), and FULL_CONTROL (having full control permissions, including READ, WRITE, READ_ACP, and WRITE_ACP).

The object service adopts Access Key IDs (AKs) and Secret Access Keys (SKs) to authenticate user identities. Keyed-hash message authentication code (HMAC) calculation is performed during authentication. The HMAC calculation uses hash algorithms to generate a message digest after inputting an SK and a message.

Each client user has an AK and an SK. The AK is used to uniquely identify the user, and the SK is used to calculate the signature. The SK is encrypted before being stored. Users must properly keep their SKs and prevent SK disclosure. An operation request sent by a client contains a user AK and a signature calculated based on the user SK (HMAC uses HMAC-SHA1 or HMAC-SHA256 for signature calculation). After receiving the request, OceanStor Pacific series searches for the SK of the received AK in the system and uses the SK to calculate a signature. Then, the system compares the calculated signature with that in the request. If the two signatures are consistent, the authentication is successful. [Figure 9-5](#) shows signature calculation using an SK.

Figure 9-5 Signature calculation using an SK



9.3.1.3 HDFS Service Access Control

HDFS service access control is implemented using ACLs. An ACL defines access permissions that users have to a particular system resource. Only users who have the permissions can access the resource.

The HDFS service leverages LDAP domain authentication. Tenants can access data only after being added into LDAP domains.

The HDFS service also supports Kerberos authentication. The storage system uses the Kerberos protocol to authenticate tenant clients. Only authenticated clients can access stored data.

9.3.1.4 Block Service Access Control

Block service access control is implemented using LUN mapping mechanisms. Application servers can access LUNs only after the LUNs are mapped to the corresponding hosts.

iSCSI supports Challenge-Handshake Authentication Protocol (CHAP). To connect storage systems to physical application servers, you need to create virtual hosts and add initiators for the hosts to establish mapping relationships between the virtual hosts and physical application servers. For storage system security, you can configure CHAP authentication to control access from application servers to storage resources.

9.3.2 Data Encryption

OceanStor Pacific series supports two types of static data encryption: self-encrypting disks (SEDs) and software-based encryption, protecting data stored in storage systems from being disclosed.

- SEDs

You can configure SEDs and internal key managers and external key management servers to encrypt data on disks for both the structured (block) and unstructured (file, object, and HDFS) services. SEDs provide two-layer security protection by using authentication keys (AKs) and data encryption keys (DEKs).

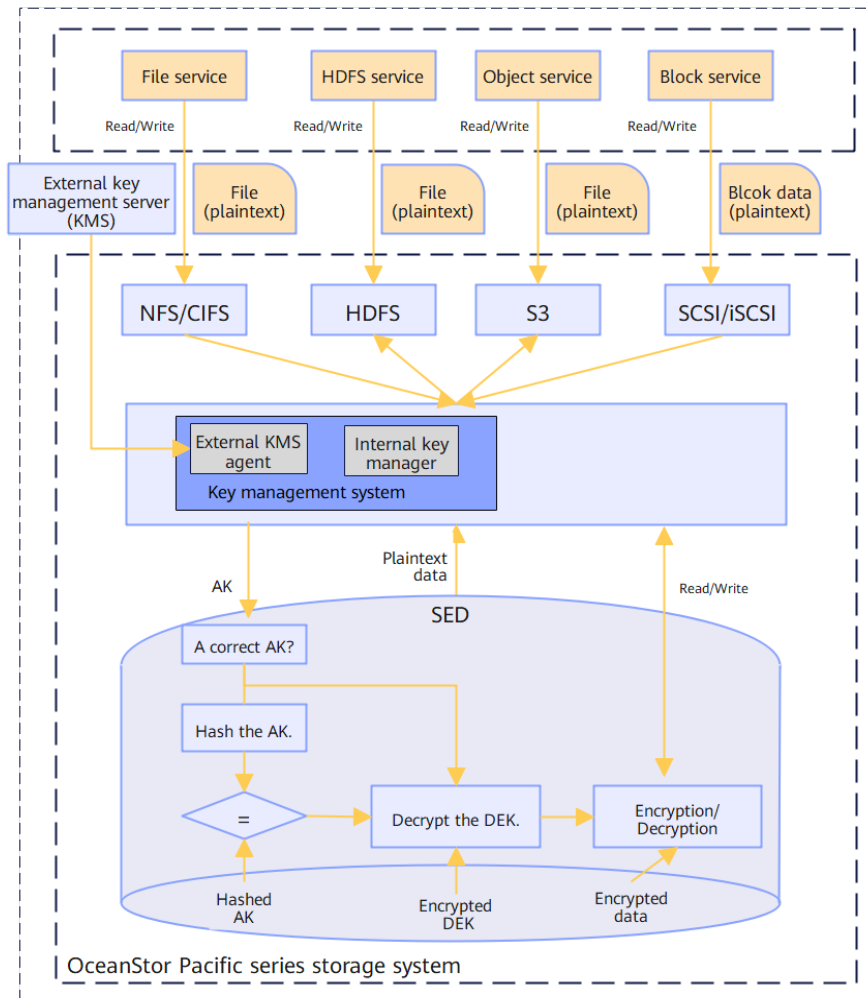
- Two-layer key management
 - AK: AKs are used for identity authentication and DEK encryption and decryption during disk access. AKs are managed by internal key managers. Updating AKs does not affect encrypted data.
 - DEK: DEKs are preset in disks. DEKs use AES-256-XTS to encrypt and decrypt service data. To destroy data, just update DEKs.
- Key management system (KMS)

Supports both internal and external key management.

 - External key management server: complies with the standard KMIP and supports KMIP 1.0, KMIP 1.1, and KMIP 1.2.
 - Internal key manager: the built-in key management application of a storage system. It manages the lifecycle of AKs, featuring easy deployment, configuration, and management. AKs are generated as secure random numbers. To implement key security protection, a three-layer architecture is adopted, and key backup, restoration, update, and destruction are supported.
- Encryption engine

The encryption engine of an SED provides basic capabilities for data encryption and decryption.

Figure 9-6 Principle of an SED



- **Software-based encryption**

Software-based encryption supports both structured data (block service) and unstructured data (file, HDFS, and object services). Two encryption algorithms are supported: XTS-AES-128 and XTS-AES-256. Software-based encryption encrypts static data using an Internal Key Manager and a built-in encryption engine and provides account-level security through two-layer key management. After data is written to a storage system, the data is encrypted at the software layer and then written to disks for storage. When the data is read, the ciphertext data is read from the disks, decrypted at the software layer to plaintext, and then provided to users.

- **Two-layer key management**

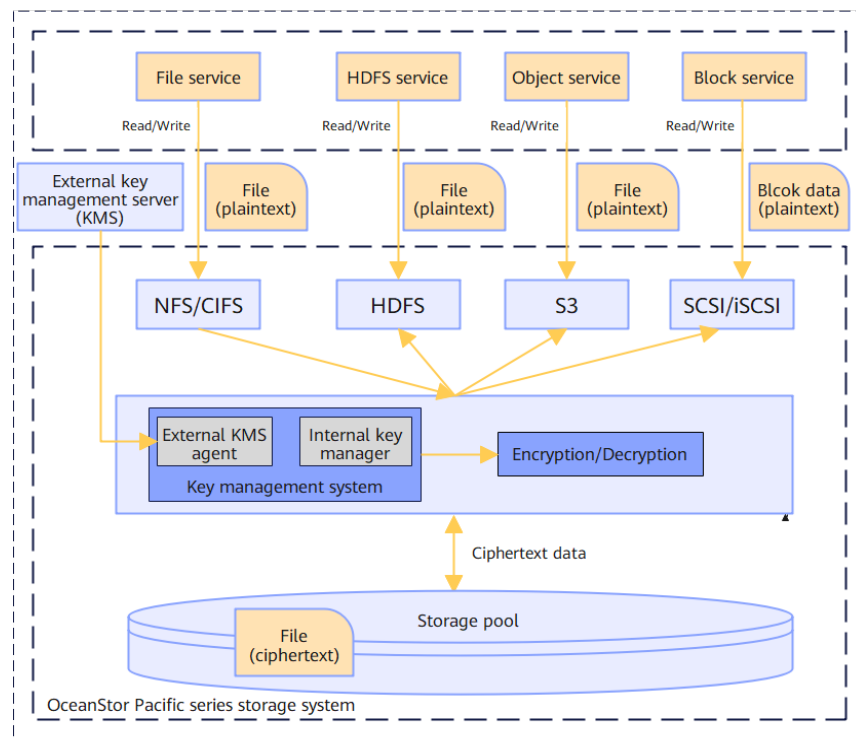
- AK: configured on tenants for identity authentication and can be updated
- DEK: used to encrypt and decrypt user data

- **KMS**

Supports both internal and external key management.

- External key management server: complies with the standard KMIP and supports KMIP 1.0, KMIP 1.1, and KMIP 1.2.
- Internal key manager: the built-in key management application of a storage system. It manages the lifecycle of AKs, featuring easy deployment, configuration, and management. AKs are generated as secure random numbers. To implement key security protection, a three-layer architecture is adopted, and key backup, restoration, update, and destruction are supported.
- Encryption engine
A built-in encryption engine is used to provide basic capabilities for data encryption and decryption. After data encryption is enabled, the built-in encryption engine encrypts data when the data is written and decrypts data when the data is read.

Figure 9-7 Principle of software-based encryption



9.3.3 Service Access Log Audit

OceanStor Pacific series provides the SmartAuditlog feature for unstructured services. It enables you to configure the audit switch and operation word on a namespace to record the file or directory operations on the namespace.

9.3.4 WORM

Write once read many (WORM) means that a file can be written once and read multiple times. Users can set a protection period for the file. During the protection period, the file can only be read but cannot be modified or deleted. After the

protection period expires, the file can be deleted or read. Therefore, WORM is a mandatory feature for a file system to serve as an archive system. WORM consists of enterprise-level and regulation compliance WORM. The administrators of enterprise-level WORM can delete files protected by WORM during the protection period, but the files protected by regulation compliance WORM cannot be deleted before the protection period expires. The unstructured services of OceanStor Pacific series support enterprise-level WORM and compliance WORM. The object service is compatible with the Amazon S3 Object Lock interface.

Generally, the WORM feature is enabled in the following process:

- **Enabling a WORM clock**
A WORM clock is used to time WORM files. After a WORM clock is set, the system times protection periods according to the clock. This ensures that files are properly protected even if the local clock time is changed.
Each directory or file has the atime, ctime, and mtime properties. After the WORM properties are set for a directory or file, the directory or file uses the WORM clock to count the time, preventing changes in the protection period of the directory or file due to changes in the local time of a node.
A WORM clock can only be set once, and cannot be modified after being set.
- **Configuring a WORM policy**
OceanStor Pacific series allows you to configure WORM policies for namespaces or dtrees. A WORM policy includes a default retention period and allows you to set a default retention period for a specific file or files with a name prefix. If the default retention period is enabled, uploaded files will be automatically retained according to the default retention period. If no default retention period is set, uploaded files will not be protected by default.
- **Accessing a WORM file**
When accessing a WORM file, a storage service checks whether the retention period of the file expires based on the WORM clock of the system instead of the local system time. Common users have only the read permission. If the retention period does not expire, the file cannot be overwritten or deleted. If the retention period expires, the file can only be deleted or read, but cannot be modified. OceanStor Pacific series supports enterprise-level WORM. Therefore, privileged users can perform all operations such as deletion and modification during the retention period.
- **Recording a WORM audit log**
OceanStor Pacific series supports the WORM audit log function. When operations are performed on files protected by WORM, special audit logs are recorded to facilitate audit by administrators. The WORM audit log function is disabled by default and needs to be enabled manually.

9.4 Storage Management Security

9.4.1 Authentication

The system supports local user authentication. Local user authentication is required before system login.

The system also supports authentication with external AD/LDAP domain authentication servers.

9.4.2 Role Management

- Access control using roles: The system provides built-in roles to implement access control.
The built-in roles include:
 - Super administrator: has full administrative permissions on storage devices and can create users of any roles.
 - Administrator: has the permission to configure and view the Call Home service (interconnection with cloud) and to view user security policies and alarms.
 - System viewer: has the permission to view alarms, security policies, licenses, and the Call Home service.
 - SAN resource administrator: has the permission to configure and manage storage pools and volumes.
 - Security administrator: has the permission to view users and configure system security, including managing security rules, security policies, certificates, and Key Management CBB (KMC).
 - Machine-machine account: internal machine-machine user of the system.
- Access control using IP address trustlists: The system supports access control by configuring IP address trustlists.

9.4.3 Log and Alarm Management

- Log management
All operations performed on management interfaces are recorded in logs. Logs detail event generation time, user IDs (including associated terminals, ports, network addresses, or communication devices), event types, names of accessed resources, and event results. Logs can be queried. When the log storage space is used up, the system automatically dumps and deletes logs. Log time is synchronized with a unified time source.
- Alarm management
Anomalies and faults occurring in the system are displayed on management interfaces in real time, prompting you to handle them in a timely manner. You can set email, Syslog, and Trap alarm notification, configure SNMP security policies, and dump alarms to independent alarm servers through SFTP.

9.4.4 Web Security

OceanStor Pacific series hardens web security using the following techniques:

- Automatic redirection of user requests to HTTPS links
DeviceManager can automatically redirect user requests to HTTPS links. When users access DeviceManager through HTTP, DeviceManager automatically redirects the access requests to HTTPS links to enhance security.
A self-signed certificate is provided for HTTPS access. You are advised to replace the certificate with a CA certificate issued by a trustworthy certificate authority.

- Prevention against cross-site scripting (XSS)
XSS attacks occur when attackers use a vulnerable web application to send malicious code to users. To prevent XSS attacks, web security configurations, such as HttpOnly, have been set for storage systems.
- Prevention against SQL injection
SQL injection is a code injection technique. Malicious SQL statements are inserted into an entry field of a web form or into a query string of a page request for execution. To prevent SQL injection, strong verification is implemented for input SQL statements.
- Prevention against cross-site request forgery (CSRF)
If a user logs in to website A and then to website B (containing attack programs) before the session on website A expires, an attacker can obtain the session ID of website A and log in to website A to intercept critical information of the user. To prevent CSRF, token information is added into message headers.
- Protection for sensitive information
Sensitive information is hidden to prevent interception by attackers.
- Restriction on file upload and download
File upload and download are restricted to prevent mission-critical files from being disclosed and insecure files from being uploaded.
- Prevention against unauthorized uniform resource locator (URL) access
Each user type is granted specific permissions, and users cannot access data beyond their permissions.

9.4.5 User Security

User security is implemented mainly using login policies. [Table 9-2](#) describes the parameters.

Table 9-2 User login policy

Parameter	Description	Value
Session Timeout Duration (Minutes)	Default session timeout period.	[Default value] 30
Account Lockout	If this parameter is enabled, a user will be automatically locked when the number of consecutive incorrect password attempts reaches Lockout Threshold within 5 minutes.	[Default value] Enabled

Parameter	Description	Value
Lockout Threshold	Maximum number of allowed consecutive incorrect password attempts. A user will be automatically locked when the number of consecutive incorrect password attempts reaches Lockout Threshold within 5 minutes.	[Value range] Integer from 1 to 9 [Default value] 3
Lockout Mode	Mode of automatically locking a user. - If you select Permanent, users will be locked permanently, and super administrators will be automatically unlocked after 15 minutes. - If you select Temporary, you can set the duration to lock users.	[Default value] Temporary
Automatic Unlock in (Minutes)	Time when the system automatically unlocks a user. - This parameter is available only when Account Lockout is enabled and Lockout Mode is Temporary. - This parameter takes effect only for users automatically locked by the system. Users who are manually locked can only be manually unlocked. - This parameter takes effect only for users except for super administrators. Super administrators will be automatically unlocked after 15 minutes no matter whether you select Permanent or Temporary.	[Value range] Integer from 3 to 2000 [Default value] 15

9.4.6 Password Security

The system supports strong password complexity policies to prevent brute force password cracking. Passwords are encrypted before being stored or transmitted. A password can be changed only after user authentication has been completed. Except for super administrators, users can change their own passwords only. [Table 9-3](#) describes the password policy.

Table 9-3 Password policy

Parameter	Description	Value
Max. Length	Maximum length of a password, preventing you from setting overly long passwords.	[Value range] Integer from 8 to 32 [Default value] 16
Min. Length	Minimum length of a password, preventing you from setting overly short passwords.	[Value range] Integer from 8 to 32 [Default value] 8
Complexity	Complexity of a password, preventing you from setting overly simple passwords.	[Value range] A password must contain special characters and at least two of the following types: uppercase letters, lowercase letters, and digits or A password must contain special characters, uppercase letters, lowercase letters, and digits [Default value] A password must contain special characters and at least two of the following types: uppercase letters, lowercase letters, and digits
Duplicate Characters	Maximum number of consecutive duplicate characters allowed in a password.	[Value range] Integer from 0 to 9. Value 0 indicates unlimited. [Default value] 3
Retained Historical Passwords	Number of historical passwords retained for a user. A new password must be different from any of the historical passwords. Value 0 indicates unlimited.	[Value range] Integer from 0 to 30 [Default value] 3

Parameter	Description	Value
Password Validity Period (Days)	Whether to set password validity. You are advised to enable this parameter. After Password Validity is enabled, you need to specify the number of days during which a password remains valid. After the validity period of a password expires, the system prompts you to change the password.	[Value range] Integer from 1 to 999 [Default value] 90
Password Expiration Warning Period (Days)	Number of days prior to password expiration that a warning about password expiration is displayed.	[Value range] Integer from 1 to 99 [Default value] 7
Password Change Interval (Minutes)	Minimum lifespan of a new password before it can be changed.	[Value range] Integer from 1 to 9999 [Default value] 5

10 Ecosystem Compatibility

OceanStor Pacific series supports various interfaces, such as NFS, SMB, HDFS, S3, and iSCSI. In addition, OceanStor Pacific series is compatible with mainstream open interfaces, and can be smoothly integrated with cloud data centers and application platforms.

[10.1 Ecosystem Compatibility of the Data Plane](#)

[10.2 Ecosystem Compatibility of the Management and Control Plane](#)

10.1 Ecosystem Compatibility of the Data Plane

10.1.1 Compatibility with Storage Protocols

Storage services provided by OceanStor Pacific series comply with mainstream protocols and standards in the industry.

- Block service interfaces: comply with the standard SCSI semantics and iSCSI protocol and provide scale-out block storage services.
- File service interfaces: comply with the standard NFS and SMB protocols and DPC that supports POSIX semantics, and provide scale-out file storage services.
- Object service interfaces: comply with the Amazon S3 protocol and are compatible with all major Amazon S3 functions.
- HDFS service interfaces: comply with the native HDFS protocol to provide Hadoop platform capabilities.

10.1.2 Block Service Compatibility with Virtualization Platforms

OceanStor Pacific series provides scale-out block data storage services, supports standard SCSI and iSCSI storage interfaces, and can interconnect with mainstream virtualization platforms, including VMware, Xen, KVM, Hyper-V, and Huawei FusionSphere. Huawei also provides detailed best practices for VMware and Hyper-V.

For details about the compatibility, visit Huawei Storage Interoperability Navigator at:

<https://info.support.huawei.com/storage/comp/#/home>

10.1.3 Block Service Compatibility with Databases

The block service of OceanStor Pacific series supports mainstream databases, including Oracle, IBM Db2, Sybase IQ, and Dameng. Huawei also provides active-active best practices of block services for Oracle RAC.

For details about the database compatibility, visit Huawei Storage Interoperability Navigator at:

<https://info.support.huawei.com/storage/comp/#/home>

10.1.4 Block Service Compatibility with Operating Systems

In addition to providing storage services for virtualization platforms, OceanStor Pacific series provides SCSI storage services by deploying OceanStor Pacific series storage drivers in operating systems of physical servers.

For mainstream operating systems, OceanStor Pacific series provides complete compatibility verification and releases compatibility certification descriptions. The compatibility will be updated when each version is released.

OceanStor Pacific series also supports operating systems that are not included in the compatibility list on the website. However, compatibility certification tests are required.

For details about the compatibility, visit Huawei Storage Interoperability Navigator at:

<https://info.support.huawei.com/storage/comp/#/home>

10.1.5 File Service Compatibility

The file service of OceanStor Pacific series provides standard NFS, SMB, FTP/FTPS, POSIX, and MPI-IO interface protocols to interconnect with mainstream compute servers and provide standard NAS protocol-based access capabilities. The compatibility of DPC that provides the POSIX and MPI-IO interface protocols is subject to the product compatibility list. The operating systems running on compute nodes that are not included in the compatibility list must pass the product adaptation and compatibility tests before being added to the compatibility list of DPC.

For details about the compatibility, visit Huawei Storage Interoperability Navigator at:

<https://info.support.huawei.com/storage/comp/#/home>

10.1.6 Object Service Compatibility

The object service of OceanStor Pacific series provides standard Amazon S3 protocol interfaces and is compatible with mainstream backup, archiving, and third-party software, such as Veritas NetBackup, Commvault Simpana, and EISOO.

For details about the compatibility, visit Huawei Storage Interoperability Navigator at:

<https://info.support.huawei.com/storage/comp/#/home>

10.1.7 HDFS Service Compatibility

The HDFS service of OceanStor Pacific series provides standard HDFS interfaces to interconnect with mainstream big data application platforms in the industry. It supports interconnection with big data products from mainstream vendors, such as Huawei FusionInsight, Cloudera CDH, Hortonworks HDP, and Transwarp TDH.

For details about the compatibility, visit Huawei Storage Interoperability Navigator at:

<https://info.support.huawei.com/storage/comp/#/home>

10.2 Ecosystem Compatibility of the Management and Control Plane

10.2.1 Integrated Network Management Platforms

IT O&M management platforms play a crucial role in data centers. They manage IT data centers in a unified manner and enable convenient device status management, monitoring, and configuration.

OceanStor Pacific series supports standard SNMP v2 and v3 management protocols and provides open REST APIs to implement out-of-band centralized access management and unified maintenance.

10.2.2 OpenStack Integration

OpenStack aims to provide open-source software to facilitate the construction and management of public and private clouds. It is an open cloud management platform in the industry, and covers computing, storage, networks, and the like. OpenStack uses the Cinder and Manila modules to manage block and file services respectively.

OceanStor Pacific series provides scale-out block, object, and file services, provides standard Cinder and Manila drivers for the interconnection with OpenStack, and uses REST APIs to manage storage resources. Huawei has released providers for each OpenStack storage module to interconnect with mainstream release versions and commercial versions of OpenStack. OceanStor Pacific series is compatible with the OpenStack R version and will be compatible with later versions.

10.2.3 Compatibility with Container Platforms

OceanStor Pacific series releases CSI Plugin for the structured data storage service and supports mainstream container management platforms Kubernetes and Red Hat OpenShift.

11

Mappings Between Features and Services

Feature	Service
SmartMulti-Tenant	File, object, and HDFS
SmartQuota	File, object, and HDFS
SmartTier	File, object, and HDFS
SmartEqualizer	File, object, and HDFS
SmartIndexing	File, object, and HDFS
SmartTakeover	HDFS
SmartQoS	File, object, HDFS, and block
SmartAuditlog	File, object, and HDFS
SmartEncryption	File, object, HDFS, and block
SmartDedupe and SmartCompression	Block
vVol	Block
HyperSnap	File, object, HDFS, and block
HyperReplication	File, object, HDFS, and block
HyperMetro	Block
HyperGeoMetro	Object
HyperGeoEC	Object
HyperClone	Block
Recycle Bin	File, object, and HDFS

Feature	Service
Scenario-specific SmartCompression	File, object, and HDFS
Standard SmartCompression	File, object, and HDFS
SmartMove	Block
SmartMigration	File and block

12 Abbreviations, Acronyms, and Glossary

Acronym/ Abbreviation	Full Spelling
AD	active directory
CA	client agent
CLI	command-line interface
CMS	cluster management service
DAS	direct-attached storage
DHT	distributed hash table
DNS	domain name system
DPC	Distributed Parallel Client
DS	data service
Dtree	directory tree (a special directory)
EC	erasure coding
FTP	File Transfer Protocol
FTPS	File Transfer Protocol Secure
GID	group ID
GUI	graphical user interface
HTTP	Hypertext Transfer Protocol
IAM	Identity and Access Management
IB	InfiniBand
IPMI	Intelligent Platform Management Interface

Acronym/ Abbreviation	Full Spelling
KV	key value
LDAP	Lightweight Directory Access Protocol
LUN	logical unit number
LVS	Linux Virtual Server
MDC	MetaData Controller
MPI	Message Passing Interface
NAS	network-attached storage
NFS	network file system
NIS	network information service
NS	Namespace
NUMA	non-uniform memory access (an architecture model for memory access from multiple CPUs)
OAM	operation, administration, and maintenance
OSD	Object Storage Device
Paxos	A highly fault-tolerant consistency algorithm based on message transfer
plog	persistence log
POSIX	Portable Operating System Interface
RAID	redundant array of independent disks
RDMA	remote direct memory access
SAN	storage area network
SAS	serial attached SCSI
SATA	Serial Advanced Technology Attachment
SCSI	Small Computer System Interface
SMB	Server Message Block (a client/server and request/response protocol)
SSD	solid-state drive
TCP	Transmission Control Protocol
UID	user identity
ZK	ZooKeeper