FusionStorage HDFS for Big Data

Technical White Paper

Issue	**01**
Date	**2019-06-10**

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions

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

Email: support@huawei.com

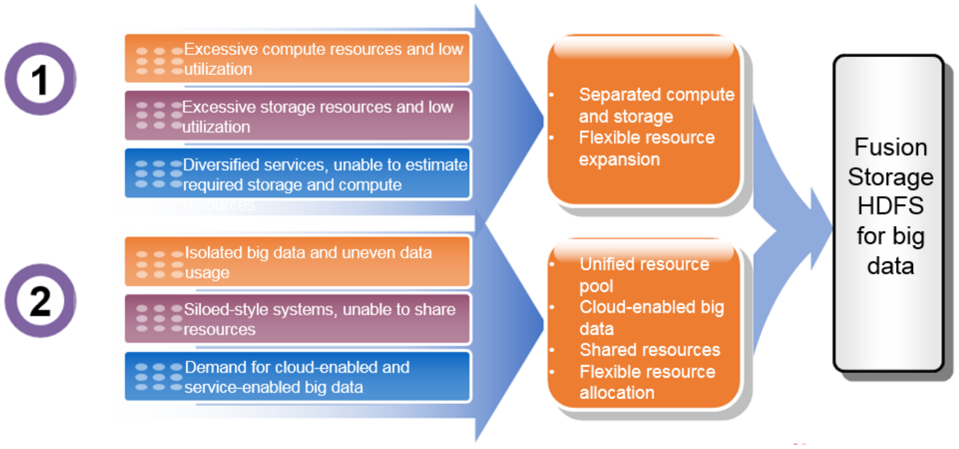
Contents

1 Overview

Nowadays, data increases exponentially in fields, such as scientific research, medical insurance, banking, government, Internet, Smart City, and carrier. Technologies, such as Internet, Internet of Things (IoT), and artificial technologies (AI), are ever-changing. All these require storage and analysis of massive amounts of data. Data has penetrated in every industry and field and become a critical production factor. A large amount of data is waiting for mining and analysis to support new business growth.

During the use of big data clusters, many pain points emerge and become even pressing with the increase in data and services. New challenges inevitably nurture new requirements. Huawei FusionStorage HDFS for big data is developed to address the requirements.

Figure 1-1 Industry application pain points


1
Excessive compute resources and low utilization
Excessive storage resources and low utilization
Diversified services, unable to estimate required storage and compute resources
2
Isolated big data and uneven data usage
Siloed-style systems, unable to share resources
Demand for cloud-enabled and service-enabled big data
Separated compute and storage
Flexible resource expansion
Unified resource pool
Cloud-enabled big data
Shared resources
Flexible resource allocation
Fusion Storage HDFS for big data

Huawei FusionStorage HDFS for big data is a distributed storage product with extensive scale-out capabilities and provides enterprise-class reliability and availability. It allows big data clusters to be flexibly and elastically expanded as simple as stacking building blocks and makes full use of compute and storage resources.

2 Product Highlights

FusionStorage HDFS for big data adopts a highly scalable distributed architecture to provide an efficient big data foundation and boasts the following highlights:

- **On-demand storage and compute configuration, protecting customer investments**

  FusionStorage HDFS for big data organizes storage media, such as hard disk drives (HDDs) and solid state disks (SSDs), into different types of large-scale storage pools. It separates storage from compute, achieving flexible storage and compute resource configuration, on-demand capacity expansion, and reduced investment. Because storage is separated from compute, data is removed from compute clusters. This enables the capacity of compute clusters to be expanded or reduced rapidly without data migration, implementing flexible allocation of compute resources.

- **Support for the multi-tenancy feature, helping you build unified storage resource pools**

  FusionStorage HDFS for big data allows multiple namespaces to connect to multiple compute clusters. Each compute cluster supports isolated authentication and is authenticated with its corresponding namespace in a unified manner. Storage resource pools are fully utilized through logical data isolation among namespaces, flexible space allocation, and storage capability sharing.

- **Distributed data and metadata management, elastically and effectively meeting future data access requirements**

  FusionStorage HDFS for big data adopts a fully distributed 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 native HDFS uses active and standby NameNodes and a single NameNode only supports a maximum of 100 million files. Different from the native HDFS, FusionStorage HDFS for big data adopts a fully-distributed NameNode mechanism, enabling a single namespace to support ten billions of files and the whole cluster to support trillions of files.

- **Fully compatibility with the erasure coding (EC) mechanism that is based on native HDFS semantics, helping you migrate business smoothly**

  The native HDFS EC does not support interfaces such as append, truncate, hflush, and fsync. Different from the native HDFS EC, FusionStorage HDFS for big data is fully compatible with native HDFS semantics, facilitating smooth business migration and supporting a wide range of Huawei and third-party big data platforms. FusionStorage HDFS for big data even supports the 22+2 EC scheme with a utilization rate of 91.7%, significantly higher than the utilization achieved by using the native HDFS EC and three-copy mechanism. This helps reduce your investment costs.

- **Enterprise-class storage reliability, ensuring service and data security**

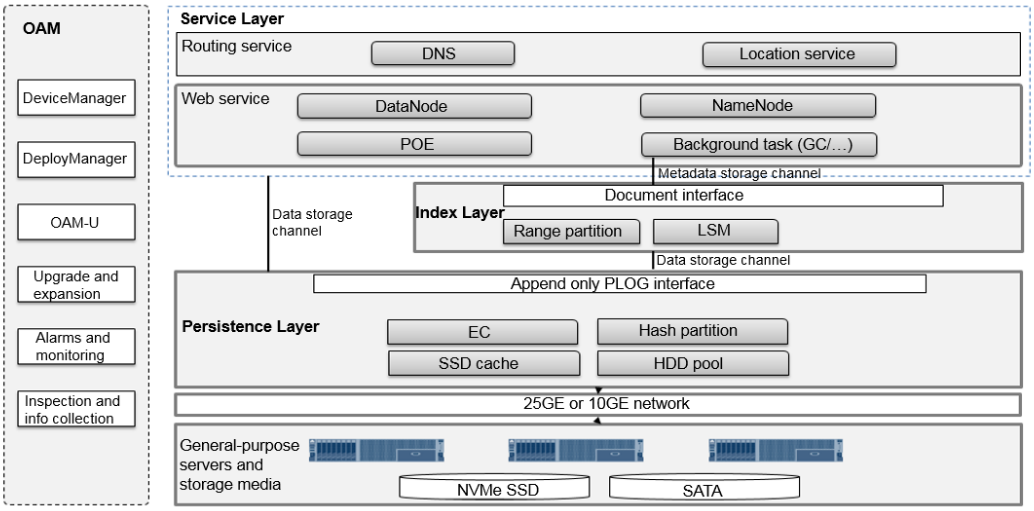
  FusionStorage HDFS for big data is developed based on the data functions virtualization (DFV) architecture, which is applied by both Huawei on- and off-cloud storage systems, providing enterprise-class storage reliability. The data reconstruction speed is as fast as 2 TB per hour, preventing data loss from subsequent faults. FusionStorage HDFS for big data supports faulty and subhealthy disk identification and fault tolerance processing, token flow control, as well as disk silence damage check, ensuring service and data security with enterprise-class storage reliability.

3 Product Architecture

3.1 Software Architecture

Huawei FusionStorage HDFS for big data is a big data storage product that supports large-scale horizontal expansion. Its software architecture complies with industry-leading scale-out, service-oriented, and microservice-based design principles.

Figure 3-1 Software architecture of FusionStorage HDFS for big data



As shown in the preceding figure, FusionStorage HDFS for big data consists of three layers: persistence layer, index layer, and service layer. These three layers are described as follows:

- The persistence layer is composed of general-purpose servers and storage media. It is responsible for data layout, load balancing, and data recovery, and provides EC data redundancy, striking a balance between performance and costs. The persistence layer is the foundation of FusionStorage HDFS for big data and determines the system scalability, performance, and reliability.
- The index layer is responsible for metadata distribution, indexing, and failover in the event of faults. It is deployed in a fully-distributed mode and provides high-speed metadata access and query capabilities for the service layer. As shown in the preceding figure, the metadata of the index layer is eventually stored in the persistence layer. Therefore, the metadata enjoys the data storage capability of the persistence layer and is evenly stored on nodes, ensuring system reliability.
- The service layer provides native HDFS APIs. It provides access to the big data storage service, a global namespace, and a variety of value-added features, such as quota and QoS. In addition, FusionStorage HDFS for big data supports the main stream HDFS protocol and implements on-demand storage resource allocation.

The software architecture of FusionStorage HDFS for big data has the following highlights:

- **Industry-leading distributed architecture**

  The distributed software architecture of FusionStorage HDFS for big data features distributed cluster management, a DHT routing algorithm, distributed stateless engines, and distributed intelligent caching. This eliminates single points of failure (SPOFs) across the whole storage system.
- **High performance and reliability**

  FusionStorage HDFS for big data balances loads among all disks and dispersedly stores data, thereby preventing data hotspots in the system. Effective routing algorithms and distributed caching ensure high performance.
- **Rapid concurrent data reconstruction**

  If disks become faulty, the system automatically, concurrently, and rapidly reconstructs the disks using data fragments distributed across different nodes in the resource pool.
- **Easy expansion and ultra-large capacity**

  The distributed stateless engines of FusionStorage HDFS for big data support ultra-large scale-out expansion, ensuring smooth and separate increases in storage and compute resources.

3.2 Data Services

FusionStorage HDFS for big data provides standard HDFS APIs. It is fully compatible with native HDFS semantics and can interconnect with a wide range of Huawei and third-party big data platforms.

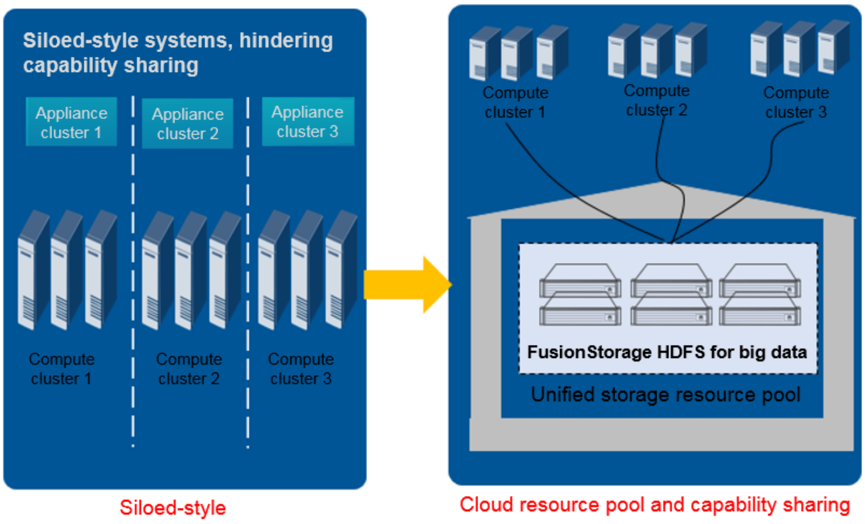
FusionStorage HDFS for big data has the following advantages:

- Adopts a cutting-edge scale-out distributed architecture and DHT routing algorithm to meet the requirements of mass data storage.
- Supports multiple services by providing external APIs compatible with the native HDFS protocol.
- Provides EC-based data protection techniques, balancing reliability and space usage.

- Supports the multi-tenant mode, making the most of enterprise and private cloud storage resources.
- Features massive scalability, high security, robust reliability, high efficiency, and wide compatibility, applicable to mass data storage and centralized backup.

3.2.1 Unified Resource Pool

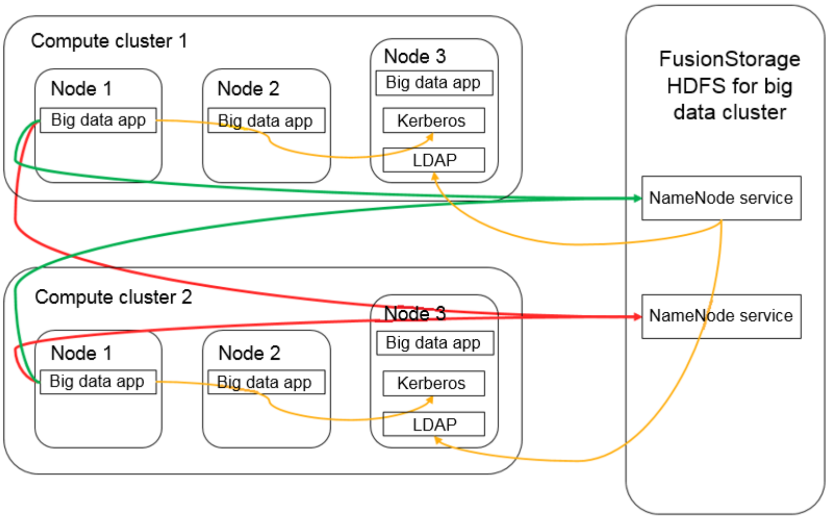
FusionStorage HDFS for big data provides a unified resource pool to connect to multiple compute clusters. FusionStorage HDFS for big data supports an independent namespace for each tenant. Storage resource pools are fully utilized through logical data isolation among namespaces, flexible space allocation, and storage capability sharing.



Transforming from siloed-style systems to unified resource pools

When using big data storage services, a tenant needs to create namespaces and create and manage data in the namespaces. The tenant can configure and modify quotas and QoS policies of namespaces. Each compute cluster supports an independent authentication system and is authenticated with the allocated namespace in a unified manner.

The following figure shows how multi-tenancy is implemented.



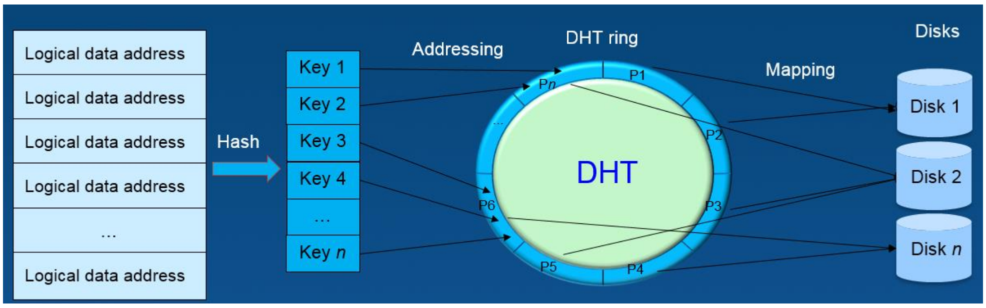
Each namespace instance replaces one HDFS service cluster to provide the same capability.

3.2.2 Distributed Hash Routing

FusionStorage HDFS for big data adopts a DHT routing algorithm to address and store data. Each storage node stores a small proportion of data.

Instead of using DHT routing algorithms, traditional HDFS storage systems manage metadata centrally. On a traditional HDFS storage system, each I/O operation will initiate a query request to the metadata service. As the system scale grows, the metadata size also increases. The concurrent operation capability of the system is subject to the capacity of the server running the metadata service. As a result, the metadata service eventually becomes a system performance bottleneck. Unlike traditional HDFS storage systems, FusionStorage HDFS for big data employs a DHT routing algorithm for data addressing, as shown in Figure 3-2.

Figure 3-2 Data addressing of FusionStorage HDFS for big data



NOTE

- **DHT ring** (also called hash space): a ring space consisting of up to 2^{32} ultra-large logical space units
- **P**: short for partition. The DHT ring is evenly divided into *N* parts (*N* indicates a number), and each part is a partition.
- **Disk**: One or more partitions map to each disk.

The DHT ring of FusionStorage HDFS for big data contains a maximum of 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 mapping during system initialization and will adjust the mapping accordingly after the number of disks in the system changes. Mapping tables occupy only a small space and are stored in the memory of the primary management node for fast routing.

The DHT ring technology adopted by FusionStorage HDFS for big data has the following advantages:

- **Outstanding performance**

  The DHT ring enables data to be evenly distributed on all disks, eliminating read and write performance bottlenecks incurred by frequent data access on certain disks. Unlike traditional HDFS storage systems, FusionStorage HDFS for big data does not manage metadata centrally. Therefore, the metadata service does not become a performance bottleneck of the system.

- **High reliability**

  The partition allocation algorithms are flexible. Identical data copies are not stored on the same disk, server, or cabinet.

- **Rapid scale-out**

  When new physical nodes are added, only part of the data needs to be migrated for load balancing.

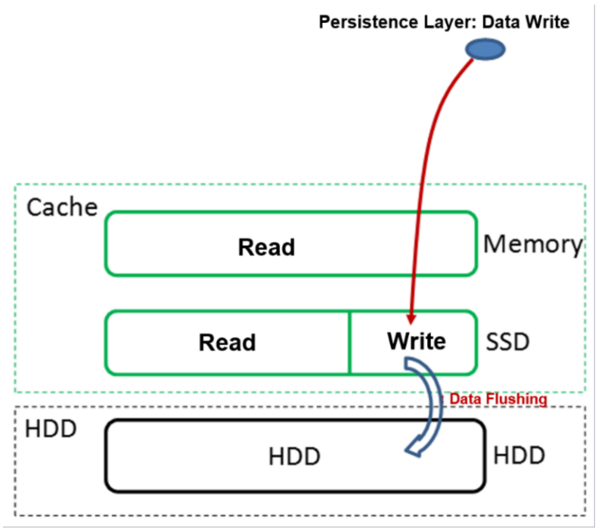
3.2.3 Cache Mechanisms

FusionStorage HDFS for big data employs a multi-level cache mechanism to improve storage I/O performance. The write and read cache mechanisms are different.

- **Write cache mechanism**

  During an I/O write on a node, the persistence stores write I/Os in the SSD cache and completes the write on the node. Then, the system periodically flushes write I/Os from the SSD cache onto HDDs in batches. A threshold is also set for the write cache. If the threshold is reached, data will also be automatically flushed to disks. Figure 3-3 shows the details.

Figure 3-3 Write cache mechanism



NOTE

FusionStorage HDFS for big data supports large I/O pass-through. By default, I/Os greater than 256 KB will be written directly to disks rather than to the cache. This configuration can be modified.

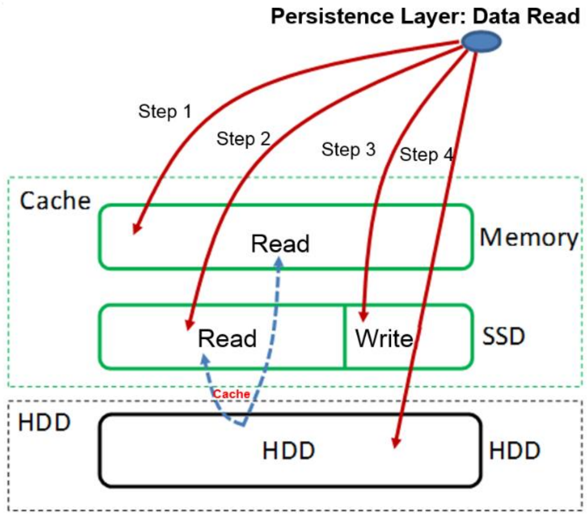
- **Read cache mechanism**

  FusionStorage HDFS for big data uses SSDs as the read cache media to speed up storage access. It employs a multi-level read cache mechanism. Level 1 (L1) is the memory cache and uses the least recently used (LRU) mechanism to cache data. Level 2 (L2) is the SSD cache and leverages the hotspot read mechanism to collect statistics of read data and record hotspot access factors. When the hotspot access factor of data reaches a specific threshold, the system automatically caches the data onto the SSD cache and removes data that has not been accessed for a long time from the SSD cache. FusionStorage HDFS for big data also supports prefetching. During a data read, FusionStorage HDFS for big data will calculate correlation of read data and fetch highly correlated data blocks to the SSD cache.

  When the persistence layer receives an I/O read operation from the upper layer:

  1. The persistence layer checks whether required I/O data is in the memory read cache. If the data is in the memory read cache, the persistence layer returns the data and moves the data to the head of the LRU queue in the read cache. Otherwise, the persistence layer proceeds to step 2.
  2. The persistence layer checks whether the required I/O data is in the SSD read cache. If the data is in the SSD read cache, the persistence layer returns the data and increases the hotspot access factor of the data. Otherwise, the persistence layer proceeds to step 3.
  3. The persistence layer checks whether the required I/O data is in the SSD write cache. If the data is in the SSD write cache, the persistence layer returns the data and increases the hotspot access factor of the data. If the hotspot access factor reaches the threshold, the persistence layer fetches the data to the SSD read cache. If the I/O data is not in the SSD write cache, the persistence layer proceeds to step 4.
  4. The persistence layer locates the required I/O data on disks and returns the data. In addition, the persistence layer increases the hotspot access factor of the data and fetches the data to the SSD read cache if the hotspot access factor reaches the threshold.

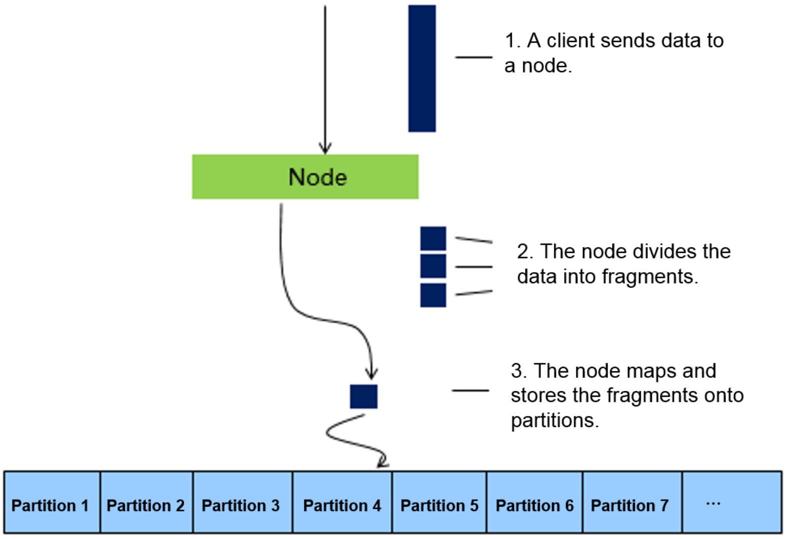
Figure 3-4 Operation procedure when the persistence layer receives an I/O read operation



3.2.4 I/O Processes

Figure 3-5 shows the data write process.

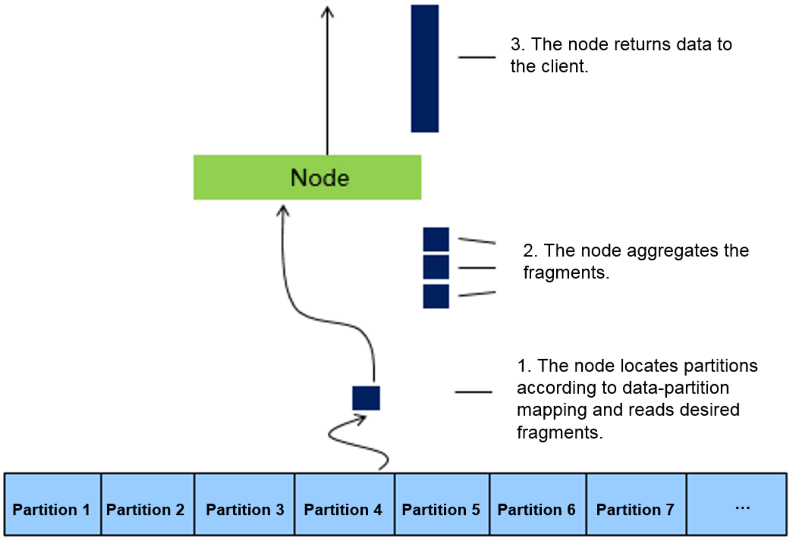
Figure 3-5 Data write process


1. A client sends data to a node.
Node
2. The node divides the data into fragments.
3. The node maps and stores the fragments onto partitions.
Partition 1
Partition 2
Partition 3
Partition 4
Partition 5
Partition 6
Partition 7
...

1. Access request: A compute node sets up a connection with a storage node and transmits data to the storage node.
2. Storage policy selection: The storage node determines the data storage policy based on user configurations.
3. Data fragmentation: The storage node calculates the fragment size based on the data storage policy and divides the data into fragments of the same size.
4. Data routing: The storage node disperses the fragments onto different disks by invoking storage APIs.

Figure 3-6 shows the data read process, which is the reverse of the data write process.

Figure 3-6 Data read process


3. The node returns data to the client.
Node
2. The node aggregates the fragments.
1. The node locates partitions according to data-partition mapping and reads desired fragments.
Partition 1
Partition 2
Partition 3
Partition 4
Partition 5
Partition 6
Partition 7
...

1. Access request: A compute node sets up a connection with a storage node and reads data from the storage node.
2. Data routing: The storage node locates partitions using the DHT routing algorithm and reads desired fragments.
3. Data restoration: If some fragments are damaged, the storage node restores them based on data storage policies.
4. Data aggregation: The storage node aggregates the fragments to generate a complete piece of data and sends the data to the compute node.

Nodes of FusionStorage HDFS for big data reserve buffers in memories to fragment and aggregate data during data writes and reads. The buffers function as follows:

- During data writes, data and parity fragments are stored in buffers and then concurrently written to multiple nodes to achieve high write efficiency.
- During data reads, storage nodes will predict the data read scope, read continuous fragments from multiple nodes in advance, and store the fragments in the buffers to improve data read efficiency.

The access service of FusionStorage HDFS for big data dynamically adjusts buffer sizes and the number of nodes that concurrently respond to reads and writes according to data sizes and connection speed of clients. This achieves the highest data throughput using the least resources.

3.2.5 Features

3.2.5.1 Data Redundancy

FusionStorage HDFS for big data implements data redundancy using EC, ensuring data reliability and availability in the event of hardware failures.

The access service of FusionStorage HDFS for big data fragments data uploaded by users, divides *N* consecutive data fragments into an EC group, and calculates the EC group using EC to generate *M* parity fragments. The data and parity fragments in each EC group are stored in a group of consecutive partitions in the storage cluster. This ensures that fragments in the same EC group are stored on different physical nodes, improving reliability.

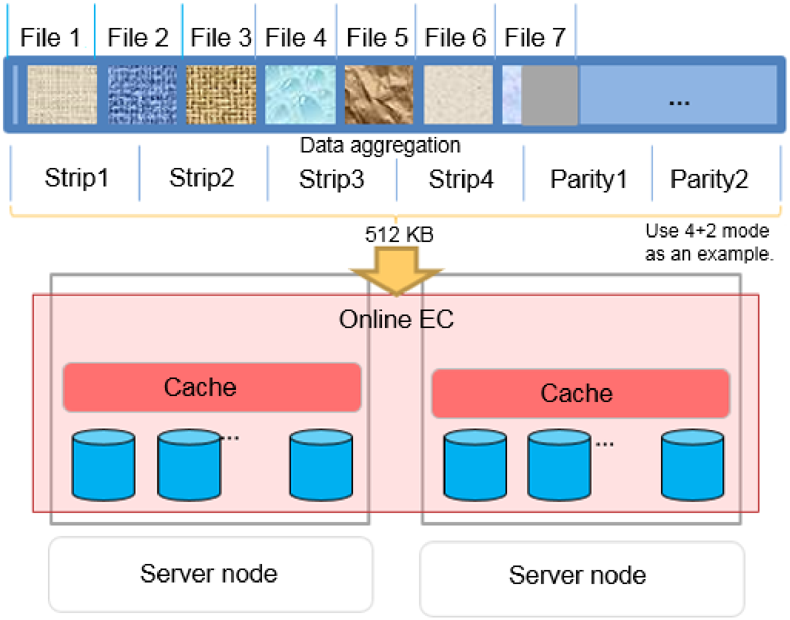
A maximum of *M* fragments can be damaged in an EC group without impacting the ability to recover a file. The access service of FusionStorage HDFS for big data can restore damaged fragments using other fragments in the EC group.

By using EC, FusionStorage HDFS for big data delivers high data reliability and provides higher storage space utilization than the multi-copy mode, striking an optimal balance between reliability and cost-effectiveness.

3.2.5.2 Online Aggregation of Small Files

The traditional HDFS storage system faces the following challenge incurred by small files: Three copies are kept for each small file and the system space utilization is only about 33% even. When the small files are erasure coded, the space utilization is still about 33% because the small files cannot occupy whole strips. To address this challenge, FusionStorage HDFS for big data provides the capability of aggregating small files online, significantly improving the space utilization. Figure 3-7 shows the aggregation process.

Figure 3-7 Online aggregation of small files

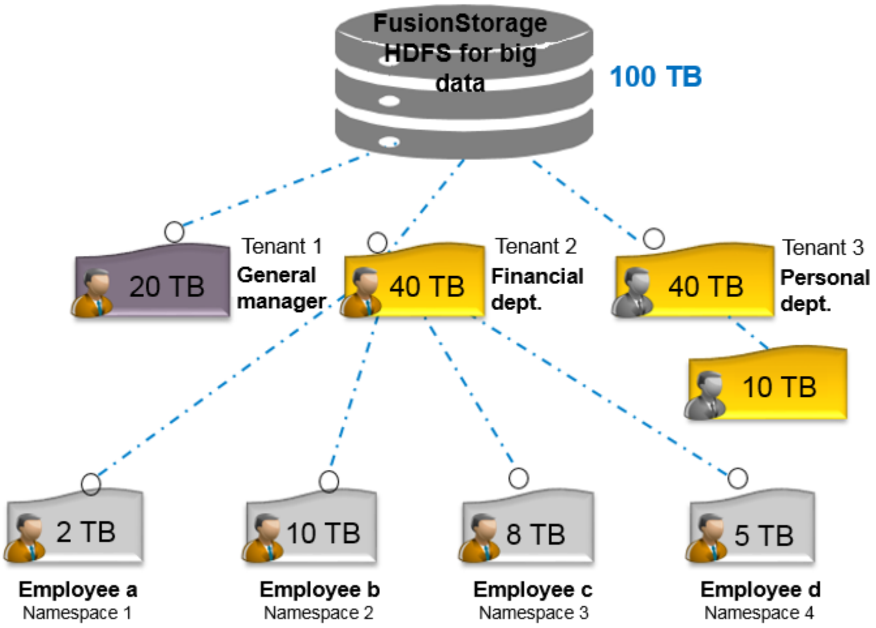


As shown in the preceding figure, small files (such as **File 1**) uploaded by clients are written into the SSD cache first. After the total size of the small files reaches the size of a stripe, the system calculates the files using EC and stores generated data fragments (such as **Strip1**) and parity fragments (such as **Parity1**) onto HDDs. In this way, small files are erasure coded, and the space utilization is significantly improved. For example, if the EC scheme is 12+3, the space utilization is about 80%, approximately 2.4 times higher than 33% of the traditional three-copy mode.

3.2.5.3 Quota and Resource Statistics

FusionStorage HDFS for big data supports namespace and tenant capacity quotas as well as resource statistics. Figure 3-8 shows a capacity quota example where company departments represent tenants and employees in the departments represent namespaces. You can set a 40 TB quota for the financial department (**Tenant 2**) and a 10 TB quota for employee b (**Namespace 2**) in the department.

Figure 3-8 Quota



The capacity quota function of FusionStorage HDFS for big data has the following characteristics:

- **Namespace capacity quota**

  Specifies the maximum size of a namespace. When the namespace size reaches the specified upper limit, new data cannot be written into the namespace.

- **Tenant capacity quota**

  Specifies the maximum capacity assigned to a tenant. When the total size of namespaces in a tenant reaches the specified upper limit, the tenant and all its users cannot write new data.

FusionStorage HDFS for big data can use REST APIs to obtain resource statistics in tenants and namespaces, such as the number and capacity of files.

- **Namespace resource statistics**

  Includes namespace sizes and the number of files in namespaces. Users can query their own namespace resources.

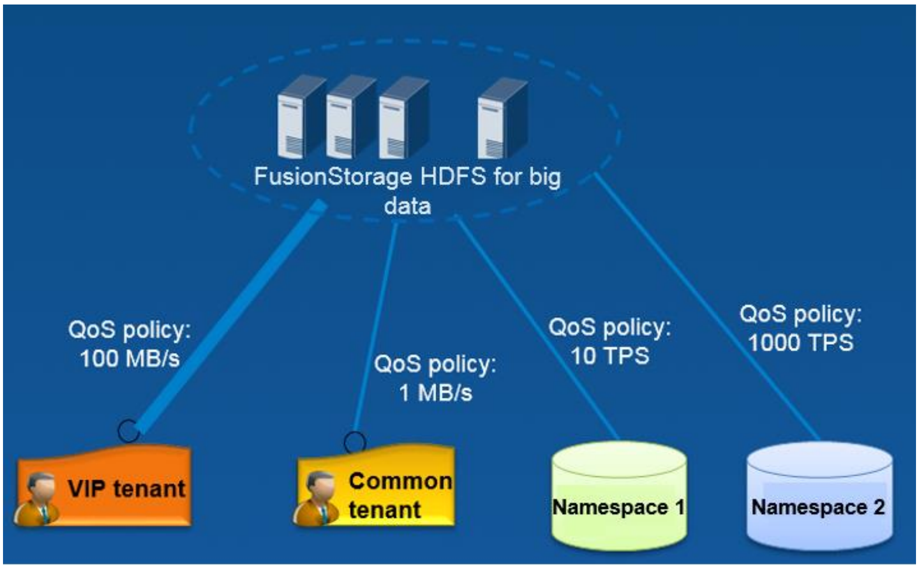
- **Tenant resource statistics**

Includes the tenant quotas, number of files in tenants, and the total capacity.

3.2.5.4 QoS

FusionStorage HDFS for big data provides QoS for the big data storage service to properly allocate system resources and deliver better service capabilities.

Figure 3-9 Tenant- and namespace-based intelligent flow control



In multi-tenant scenarios such as private cloud, customers require that transactions per second (TPS) and bandwidth resources in storage pools be properly allocated to tenants or namespaces with different priorities and that the TPS and bandwidth resources of mission-critical services be sufficient. To meet customer requirements, FusionStorage HDFS for big data provides the following refined QoS capabilities:

- **Refined I/O control**

  Enables the system to provide differentiated services for tenants and namespaces with different priorities.

- **TPS- and bandwidth-based QoS for tenants and namespaces**

  QoS allocates namespaces with different TPS and bandwidth capabilities for applications of different priorities. This maximizes storage pool resource utilization and prevents mission-critical services from being affected by other services. Different QoS policies can be configured for VIP and common tenants in the same system to ensure service quality for high-priority tenants.

3.2.5.5 Access Permission Control

FusionStorage HDFS for big data implements the same access permission control as that in the native HDFS. You can only access resources for which you have permissions. FusionStorage HDFS for big data enables namespaces and compute clusters to be authenticated in a unified manner.

3.3 Storage Management

3.3.1 Storage as a Service

As the management graphical user interface (GUI) of FusionStorage HDFS for big data, DeviceManager provides the following functions:

- **Storage pool management**

  You can create and delete storage pools, query resource statistics and disk topologies of storage pools, as well as expand or reduce capacities of storage pools.

- **Storage service configuration**

  1. Authentication configuration

     You can select an authentication mode between Provisioning Orchestration Engine (POE) and Identity and Access Management (IAM), and complete interconnection. When POE authentication is selected, you can manage service accounts.

  2. Namespace management

     You can configure namespaces and set quotas and QoS for the namespaces, as well as view the namespace list and quota usage.

3.3.2 Cluster Management

FusionStorage HDFS for big data uses cluster management software to provide the following functions:

- **Basic cluster information monitoring**

  You can query basic cluster information, including cluster names, health status, running status, versions, cluster capacities, and node quantity.

- **Performance monitoring**

  You can view the bandwidth and input/output operations per second (IOPS) of accesses.

- **Account management**

  You can create, delete, and modify storage service accounts when POE authentication is used.

- **Alarm management**

  You can configure alarm notification, as well as handle, mask, and dump alarms.

- **User management**

  You can manage users and configure security policies.

- **License management**

  You can view active licenses and import new ones.

- **Cluster management**

  You can start or stop the system, enable or disable Toolkit service, and set system time, external DNSs, and configuration file import and export rules.

- **Node management**

  You can stop and freeze nodes.

3.3.3 Cluster Expansion

FusionStorage HDFS for big data delivers superb scalability thanks to its distributed architecture. A single FusionStorage HDFS for big data cluster can contain 3 to 4096 nodes.

As the number of nodes increases, the storage and compute capabilities increase linearly. This delivers a linear growth in bandwidth and concurrent request processing capability. Capacity expansion of FusionStorage HDFS for big data has the following characteristics:

- Online capacity expansion is supported. Services are not adversely affected during capacity expansion.
- Flexible capacity expansion is delivered. Nodes can be added to existing or new storage pools.
- When nodes are added to existing storage pools, FusionStorage HDFS for big data implements rapid load balancing without migrating a large amount of data.

3.4 Recommended Hardware

FusionStorage HDFS for big data is designed based on general purpose hardware. To ensure system reliability and optimal performance, you are advised to use the hardware models listed in the following table. For more details about hardware configurations, consult your Huawei sales representative.

Hardware Type	Recommended Model	Description
Cabinet	Standard IT cabinet	Provides 42 U space for device installation.
General-purpose hardware nodes	Huawei FusionServer 5288 V5	Storage node with 36 disk slots Typical configurations: 256 GB memory, Intel Skylake 4114 V5 CPUs, and 800 GB, 1.6 TB, or 3.2 TB NVMe SSDs as cache
	Huawei FusionServer 2288H V5	Storage node with 12 disk slots Typical configurations: 256 GB memory, Intel Skylake 4114 V5 CPUs, and 800 GB, 1.6 TB, or 3.2 TB NVMe SSDs as cache
	Huawei TaiShan 5280	Storage node with 36 disk slots Typical configurations: 256 GB memory, Huawei-developed Hi1616 CPUs, and 800 GB, 1.6 TB, or 3.2 TB NVMe SSDs as cache
	Huawei TaiShan 2280	Storage node with 12 disk slots Typical configurations: 256 GB memory, Huawei-developed Hi1616 CPUs, and 800 GB, 1.6 TB, or 3.2 TB NVMe SSDs as cache
Network devices	Huawei CE6855-48S6Q-HI	10GE switch
	Huawei CE6865-48S8CQ-EI	10GE or 25GE switch

Hardware Type	Recommended Model	Description
	Huawei CE5855-48T4S2Q-EI	GE switch
Keyboard, Video, and Mouse (KVM) controller		Provides eight KVM ports.

3.5 System Networking

The network planes of FusionStorage HDFS for big data are as follows:

- **Service plane**

  Interconnects with the customer's service network and supports multiple subnets.

- **Storage plane**

  Enables communication among nodes of FusionStorage HDFS for big data and supports multiple subnets. The storage plane supports only IPv4 networking.

- **Management plane**

  Interconnects with the customer's management network, enabling maintenance terminals to access FusionStorage HDFS for big data.

- **BMC plane**

  Interconnects the Mgmt ports on nodes of FusionStorage HDFS for big data, enabling remote device management.

- **Control plane**

  Manages internal cluster information of FusionStorage HDFS for big data.

Figure 3-10 shows a networking diagram.

Figure 3-10 Networking diagram

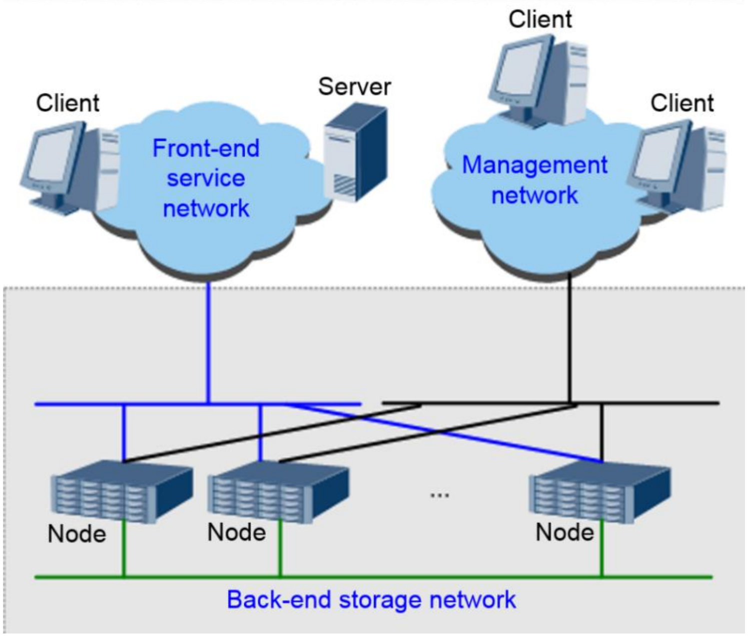


Table 3-1 lists the three networking solutions provided by FusionStorage HDFS for big data.

Table 3-1 Networking solutions

Solution	Front-End Service Network	Back-End Storage Network
10GE networking	10GE	10GE
25GE networking	25GE	25GE
GE networking	GE	10GE

In addition, FusionStorage HDFS for big data can be used in the Huawei FusionCloud private cloud solution to provide storage services. When used in the Huawei FusionCloud private cloud solution, FusionStorage HDFS for big data follows the networking principles of this solution.

3.5.1 Networking Within a Cluster

FusionStorage HDFS for big data supports two networking setups within a cluster, depending on whether service and storage planes share switches.

Figure 3-11 Networking setup in which service and storage planes use separate switches

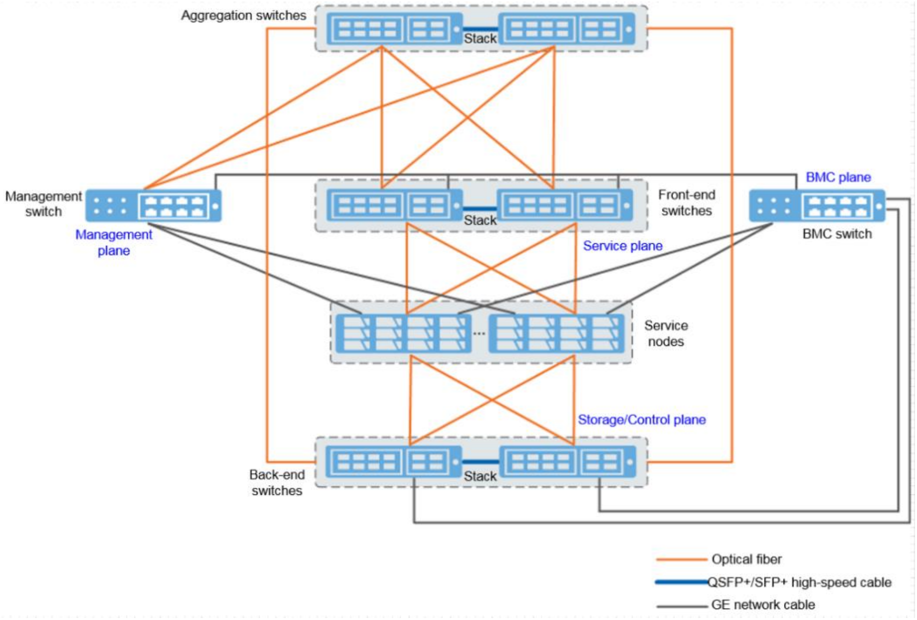
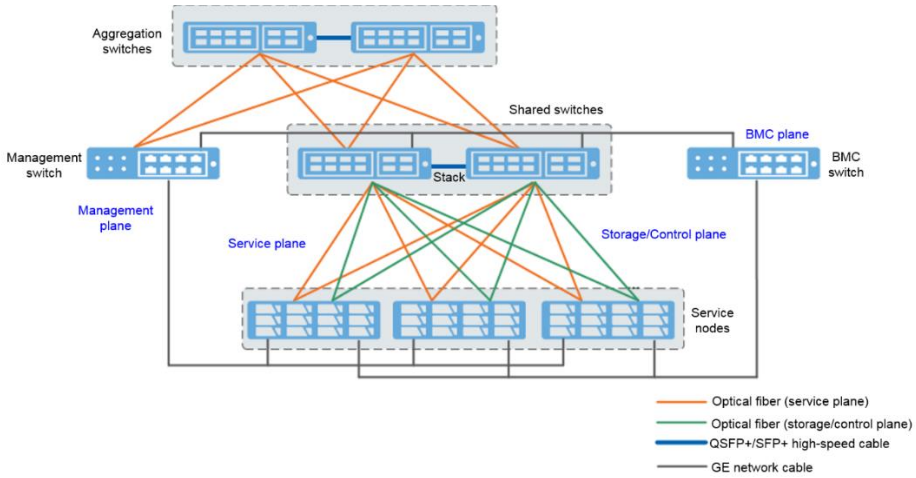


Figure 3-12 Networking setup in which service and storage planes share switches



The preceding figures show the connections between nodes and switches in a single subnet. A single cluster consists of several such subnets interconnected through aggregation switches.

3.6 DNS Deployment

3.6.1 LAN-based DNS

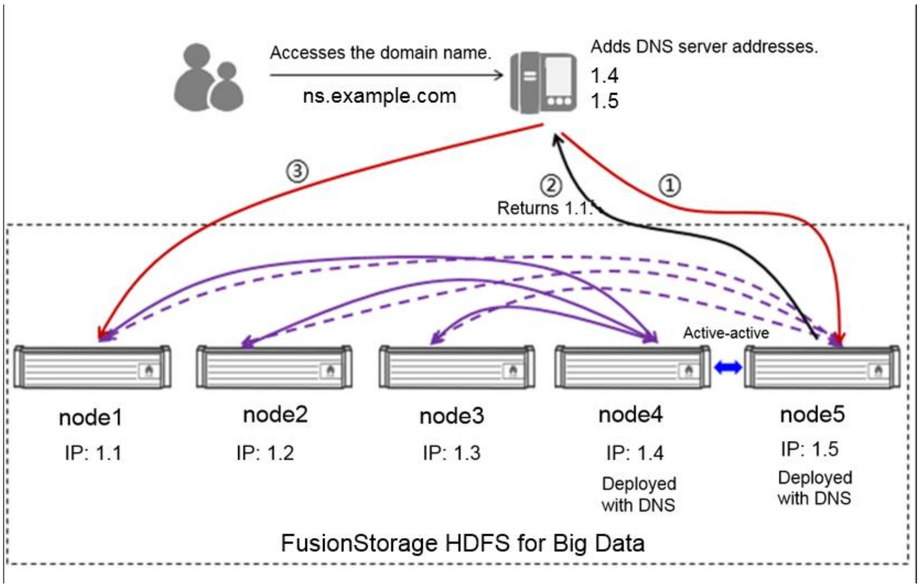
The LAN-based DNS solution is simple and convenient. This section uses a five-node cluster shown in Figure 3-13 as an example. DNS services are deployed on **node4** and **node5** and run in active-active mode.

For clarity, network switches on the storage plane are not illustrated in Figure 3-13, and **1.1**, **1.2**, **1.3**, **1.4**, and **1.5** represent external IP addresses. On compute nodes (PCs or servers), the DNS server addresses are set to the IP addresses (**1.4** and **1.5** in this example) of the nodes that run the DNS service of FusionStorage HDFS for big data.

When a user attempts to access domain name **ns.example.com**:

1. The local compute node selects one (**1.5** in this example) of the two DNS server addresses to resolve domain name **ns.example.com**.
2. The DNS service of FusionStorage HDFS for big data on **node5** resolves domain name **ns.example.com** into an IP address (**1.1** in this example) and returns the IP address to the compute node.
3. The compute node caches the IP address and then accesses the associated node (**node1** in this example). As long as the IP address is cached, the compute node does not need to request domain name resolution again. Instead, the compute node directly accesses the IP address in the cache.

Figure 3-13 LAN-based DNS solution



This solution is easy to deploy but does not support access across network segments. If the node associated with the IP address stored in the cache becomes faulty, the compute node continues to access the faulty node until the IP address in the cache expires.

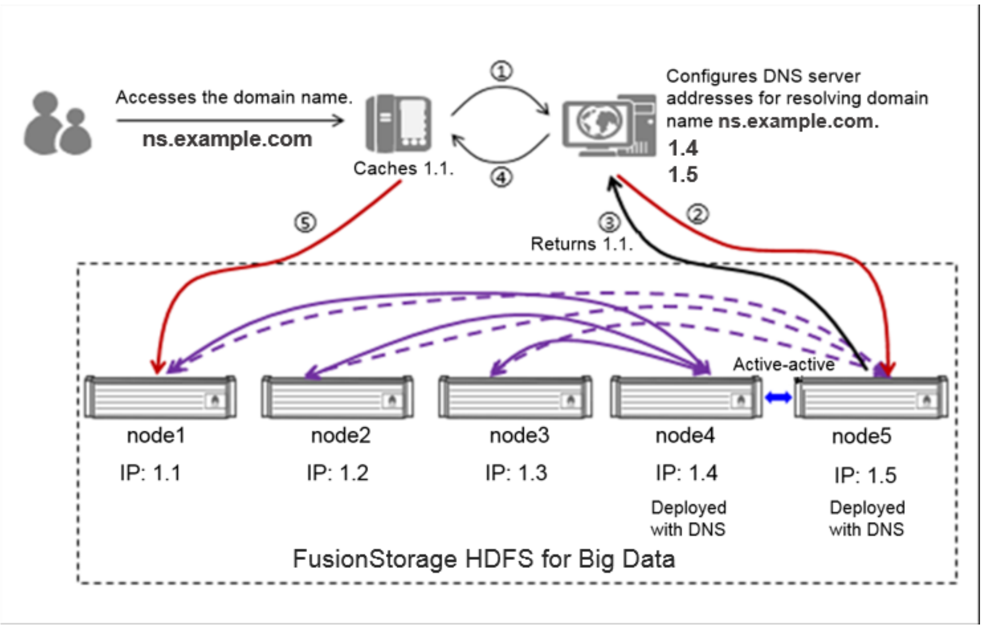
3.6.2 WAN-based DNS

Unlike the LAN-based DNS solution, the WAN-based DNS solution employs DNS servers. In the example shown in Figure 3-14, IP addresses **1.4** and **1.5** of the nodes running the DNS service of FusionStorage HDFS for big data are configured to resolve domain name **ns.example.com** on a DNS server.

When a user attempts to access domain name **ns.example.com**:

1. The compute node requests the DNS server to resolve domain name **ns.example.com**.
2. The DNS server selects one DNS server address (**1.5** in this example) from the two addresses and forwards the resolution request to the associated node (**node5** in this example).
3. **node5** resolves **ns.example.com** into IP address **1.1** (associated with **node1**) and returns the IP address to the DNS server.
4. The DNS server caches the resolved IP address locally and forwards it to the compute node.
5. The compute node caches the resolved IP address. As long as the IP address is cached, the compute node does not need to request domain name resolution again. Instead, the compute node directly accesses the IP address in the cache.

Figure 3-14 WAN-based DNS solution



The advantage of this solution is that you do not need to configure DNS information on compute nodes. Instead, you can directly use compute nodes for data access.

4 Outstanding Performance and Scalability

4.1 Superb Single-Namespace Performance

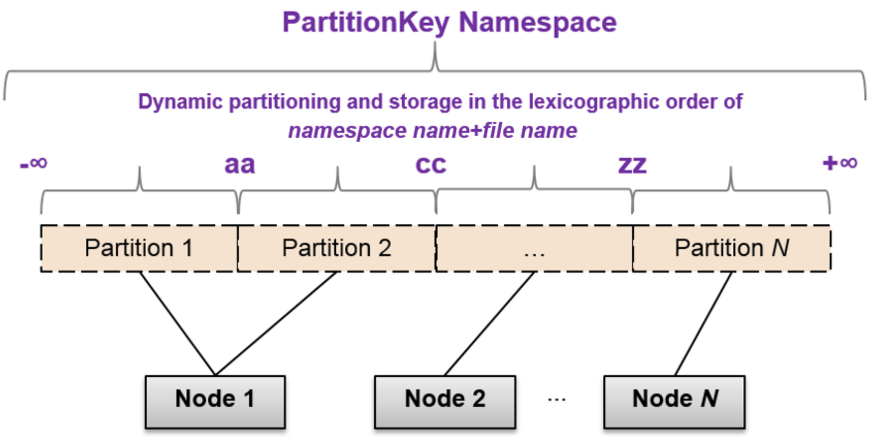
Native HDFS storage systems face the following two challenges:

- System scalability is limited, unable to meet 100 PB scalability requirements.
- Metadata access hotspots exist. The number of files a namespace can contain is limited (100 million at most).

The two challenges limit the capacity and performance of a single namespace, and the system capability cannot be fully utilized. Multiple namespaces are required to meet capacity and performance demands, increasing file storage management complexity. To address these two challenges, FusionStorage HDFS for big data improves the performance of a single namespace in the following ways:

- The service, index, and persistence layers of FusionStorage HDFS for big data are decoupled from each other and can be expanded independently. A single cluster supports up to 4096 nodes and EB-level scalability, enabling you to store, use, and manage huge volumes of data in a single resource pool. This eliminates the scalability bottleneck of a single namespace.
- Metadata is distributed using dynamic range partitioning technology. Each server only manages a group of fragmented file metadata and supports failover and dynamic load balancing.

Figure 4-1 Dynamic range partitioning



As shown in the preceding figure, FusionStorage HDFS for big data sorts file names in the lexicographic order of *namespace name*+*file name* to form a metadata collection. The metadata collection is dynamically stored in multiple partitions based on the metadata size and access frequency. The partitions storing metadata are located in different physical nodes. In this way, metadata is dispersed on all node, eliminating the bottleneck of metadata management on a single namespace.

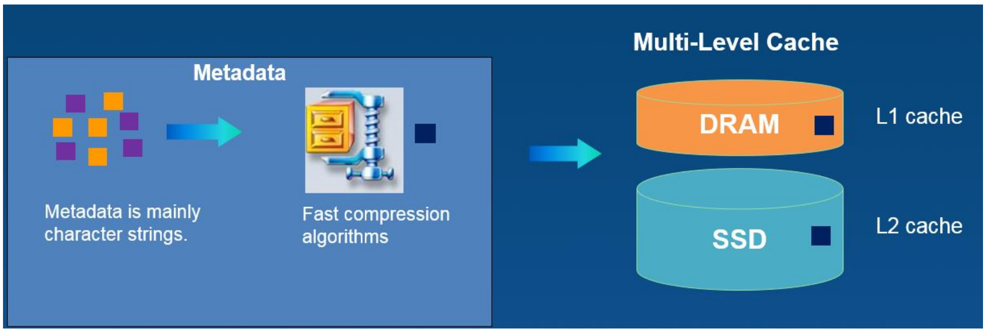
The persistence layer routes data using a DHT routing algorithm and ensures that data is evenly distributed on all nodes and disks in the system, resolving the data distribution bottleneck of a single namespace.

FusionStorage HDFS for big data supports up to 10 billion files in one namespace, fully able to meet the read and write requirements of your applications.

4.2 Multi-Level Metadata Cache

FusionStorage HDFS for big data supports multi-level metadata cache to improve the read performance of files and ensure quick access to hot data.

Figure 4-2 Multi-level metadata cache mechanism


Metadata
Metadata is mainly character strings.
Fast compression algorithms
Multi-Level Cache
DRAM
SSD
L1 cache
L2 cache

As shown in the preceding figure, metadata of FusionStorage HDFS for big data is compressed before being stored, significantly reducing the metadata volume.

- Metadata is mainly character strings, and the compression rate is high.
- Fast compression algorithms adopted by FusionStorage HDFS for big data deliver an excellent compression effect with low CPU usage.

After compression, metadata is first stored on the DRAM (L1 cache), which provides microsecond-level metadata read performance. SSDs function as L2 cache and provide millisecond-level metadata read performance.

4.3 Global Load Balancing

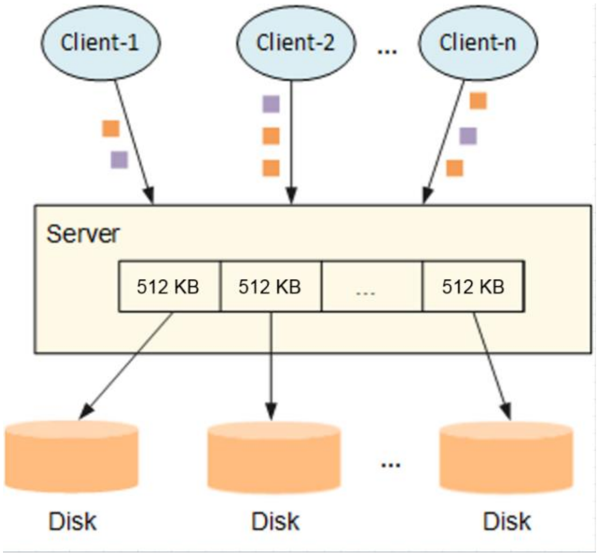
The DHT routing algorithm adopted by FusionStorage HDFS for big data evenly allocates data I/Os from upper-layer applications to disks on different servers, globally balancing load and avoiding access hotspots.

- The system automatically scatters data of each file onto different disks of multiple servers. Frequently accessed data and rarely accessed data are evenly distributed on each server, preventing hotspots in the system.
- If nodes are removed due to a failure, or if new nodes are added, FusionStorage HDFS for big data employs data restoration and reconstruction algorithms on the disks of all servers.
- Globally sorted metadata is fragmented and evenly stored on partitions of each node. The partition size is dynamically adjusted based on request frequency and total data volume.

4.4 Online Data Aggregation

FusionStorage HDFS for big data can aggregate files of different sizes into a full stripe, divide the stripe into 512 KB fragments, and write the fragments onto HDDs. This maximizes the large I/O advantages of HDDs and avoids HDD's disadvantages in IOPS.

Figure 4-3 Online data aggregation



As shown in the preceding figure, files uploaded by different clients are aggregated into 512 KB I/Os on the same server. Every *N* 512 KB I/Os are concurrently written onto *N* HDDs (EC scheme: *N*+*M*). A single HDD supports about 200 IOPS and 100 MB/s bandwidth. Assume that clients need to write 200 I/Os, each with 100 KB. If aggregation is not performed, the IOPS bottleneck of the HDD is reached but the required bandwidth is only about 20 MB/s (200 x 100 KB). If a server aggregates the 200 I/Os into forty 512 KB I/Os, the HDD only needs to provide 40 IOPS and 20 MB/s bandwidth. Neither the IOPS nor the bandwidth will constitute bottlenecks. This enables the HDD to process more I/Os, maximizing its bandwidth advantage.

4.5 Stateless Cluster

By using a one-time addressing DHT algorithm, the access service of FusionStorage HDFS for big data is loosely coupled with the storage service and nodes are stateless. Based on load balancing, any node can process service requests. The number of nodes is not limited by status synchronization or locking mechanisms, and theoretically can be infinitely increased to support linear capacity expansion.

4.6 Elastic Expansion

Elastic expansion of FusionStorage HDFS for big data has the following characteristics:

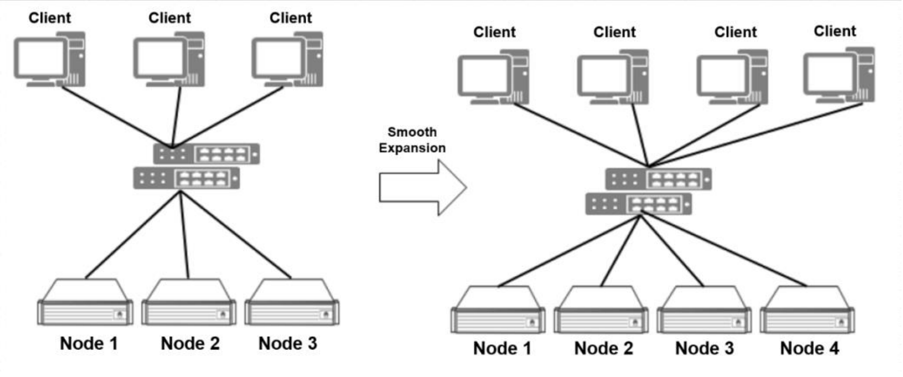
- **Fast load balancing**

  After new nodes are added, FusionStorage HDFS for big data implements fast node balancing and avoids migration of a large amount of data.
- **Flexible expansion**

  You can add disks, compute nodes, and storage nodes separately or together to expand capacity.
- **Linear performance growth**

  Compute, storage, and cache resources are evenly distributed on each node. The system TPS, throughput, and cache linearly increase as more nodes are added.

Figure 4-4 Capacity expansion of FusionStorage HDFS for big data



FusionStorage HDFS for big data supports dynamic node increase. It is recommended that a single FusionStorage HDFS for big data cluster contain 3 to 4096 nodes. As the number of nodes increases, the storage and compute capabilities increase linearly. This delivers a linear growth in bandwidth and concurrent request processing capability.

FusionStorage HDFS for big data provides a global cache whose capacity expands linearly as the number of nodes increases. In addition, more nodes result in a higher cache hit ratio of hotspot data, which greatly reduces random disk I/Os and improves the overall system performance.

Increasing capacity or performance of traditional storage systems requires horizontal expansion and reconfiguration of applications, interrupting user services. Unlike traditional storage systems, FusionStorage HDFS for big data supports minute-level capacity expansion and automatic load balancing. Modification on servers, clients, and applications is not required, and user services are not interrupted.

5 Solid Reliability

5.1 Data Redundancy Protection

FusionStorage HDFS for big data uses EC to implement data redundancy protection.

5.1.1 Data Fragmentation

To implement data protection and high read/write performance, the system performs data fragmentation. When a file is being created, the system selects suitable nodes according to the default protection level. During a data write, the system evenly distributes fragments on each selected node. During a data read, the system concurrently reads fragments from the nodes.

FusionStorage HDFS for big data stores data using EC and supports multiple data protection schemes for tenants by using different data fragmentation mechanisms. Different data protection methods are implemented based on different data fragmentation methods. Data written into FusionStorage HDFS for big data is divided into fixed-size (for example, 512 KB) data fragments. File data is split into multiple original data fragments, and 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. As long as the number of lost fragments in a stripe does not exceed *M*, data can be read and written properly. Lost fragments can be restored from the remaining fragments using a data restoration algorithm. The space utilization of the system is about *N/(N+M)*. *M* determines the data reliability and can be 2, 3, or 4. A larger value of *M* brings higher reliability.

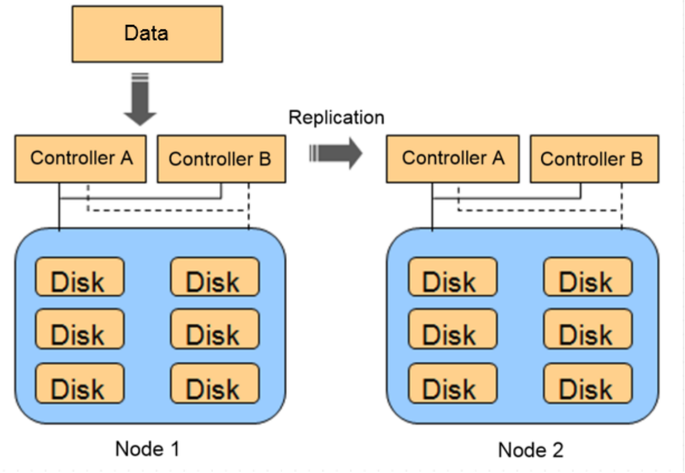
5.1.2 N+M Data Protection

FusionStorage HDFS for big data delivers higher reliability and disk utilization than storage systems that use traditional RAID technology.

The traditional RAID technology stores data on different disks in the same RAID group. If a disk fails, RAID reconstruction is implemented to restore data stored on the faulty disk. RAID levels commonly used by storage systems are RAID 0, 1, 5, and 6. RAID 6, which offers the highest reliability among all RAID levels, merely tolerates a concurrent failure of two disks at most.

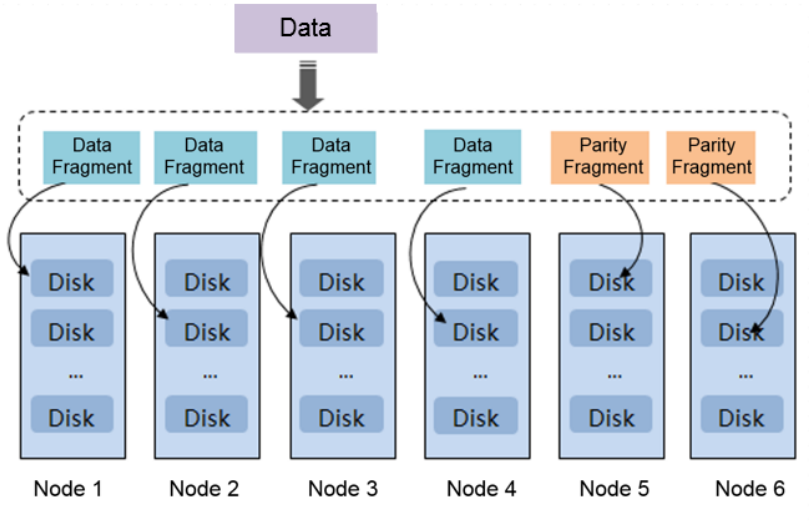
Furthermore, such storage systems use controllers to execute RAID-based data storage. To prevent controller failures, one storage system is typically equipped with two controllers to ensure service availability. However, if both controllers fail, service interruption is still inevitable. Such storage systems can further improve system reliability by implementing inter-node synchronous or asynchronous data replication, but this will decrease disk utilization, driving up TCO. Figure 5-1 shows the traditional RAID data protection.

Figure 5-1 Traditional RAID data protection



The data protection technology employed by FusionStorage HDFS for big data delivers distributed and inter-node redundancy. Data written into FusionStorage HDFS for big data is divided into *N* data fragments, and *M* parity fragments are generated for the *N* data fragments using EC. The *N*+*M* fragments are then stored on *N*+*M* nodes. Figure 5-2 shows a 4+2 protection scheme where four data fragments and two parity fragments are stored on six nodes.

Figure 5-2 N+M data protection



Because fragments of each stripe are saved on multiple nodes, FusionStorage HDFS for big data can tolerate disk- and node-level failures, ensuring zero data loss. The system functions properly as long as the number of concurrently failed nodes does not exceed *M*. Through data reconstruction, the system can restore damaged data to ensure data reliability.

The data protection schemes provided by FusionStorage HDFS for big data achieve high reliability similar to that provided by traditional RAID based on data replication among multiple nodes. Furthermore, the data protection schemes of FusionStorage HDFS for big data maintain a disk utilization of up to *N/(N+M)*. Unlike traditional RAID that requires independent hot spare disks to be allocated in advance, FusionStorage HDFS for big data allows any available space to serve as hot spare space, further improving storage system utilization.

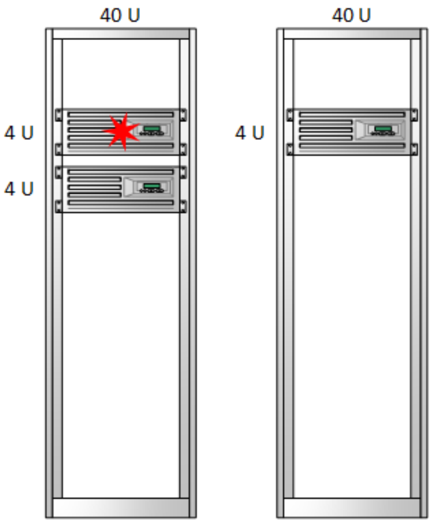
FusionStorage HDFS for big data provides multiple *N+M* data protection schemes. You can flexibly configure data redundancy on the management page to obtain your desired reliability levels.

5.1.3 Node-Level Security

FusionStorage HDFS for big data adopts a fully distributed architecture. File data and metadata are distributed on each node after fragmentation and EC. When the number of nodes is greater than or equal to *(N/M)* + 1, the system supports node-level security (if N/M is not an integer, round it up to the nearest integer). For example, if *N+M* is 4+2, only three nodes are required to implement node-level security. Each file is divided into six fragments, and each node stores two fragments. Data can still be read if any node becomes faulty for a period of time. You can use four nodes to maintain read/write performance and reliability of the 4+2 protection scheme if one node becomes permanently faulty.

Figure 5-3 shows an example of node-level security layout. If any of the nodes becomes faulty, the system will read the four fragments on the other two nodes and restore damaged fragments using EC. By doing so, node-level security is ensured. To maintain the write performance of the 4+2 protection scheme when one node is down for a long period of time, configure one more node.

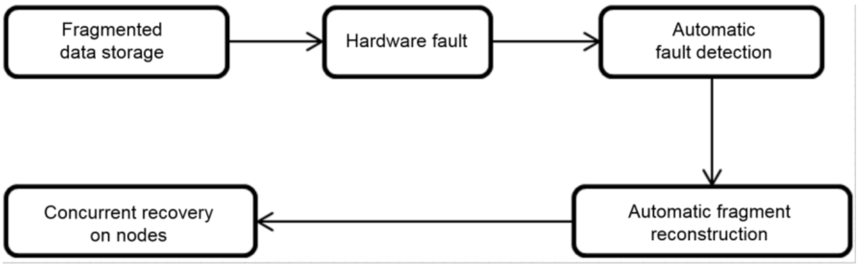
Figure 5-3 Node-level security layout



5.2 Fast Data Reconstruction

Each disk in FusionStorage HDFS for big data stores multiple data fragments whose parity fragments are distributed on other nodes in the system based on specified distribution policies. When detecting a disk or node failure, FusionStorage HDFS for big data automatically starts data recovery in the background. Because parity fragments are distributed on different nodes, data reconstruction will concurrently start on these nodes to recover data. Each node reconstructs only a small amount of data, and multiple nodes reconstruct data concurrently. This eliminates the performance bottleneck caused by the reconstruction of a large amount of data on a single node and minimizes the impact on upper-layer services. Figure 5-4 shows the automatic data reconstruction process.

Figure 5-4 Automatic data reconstruction process


Fragmented data storage
Hardware fault
Automatic fault detection
Automatic fragment reconstruction
Concurrent recovery on nodes

Concurrent and fast fault recovery and data reconstruction supported by FusionStorage HDFS for big data have the following characteristics:

- Data and parity fragments are distributed in the entire resource pool. If a disk fails, its data can be automatically and concurrently reconstructed across the whole resource pool.
- Data is distributed onto different nodes. The failure of a single node does not affect data availability and reconstruction.

- Load balancing can be automatically implemented in the event of a fault or during capacity expansion. Capacity expansion enables applications to obtain larger capacity and better performance without requiring any adjustment. The restoration of each disk in FusionStorage HDFS for big data is independent so that disks can be restored concurrently. The restoration speed is up to 2 TB/hour.

5.3 Cluster Reliability

FusionStorage HDFS for big data leverages a fully symmetric architecture. From the perspective of the physical structure, the same system software is deployed on all nodes. From the perspective of user experience, all nodes are identical and can process user requests.

The service layer provides big data storage services. When a compute node accesses a cluster system, if a single node is faulty, the compute node obtains a new IP address using the domain name resolving scheme and services are automatically switched to other nodes.

To ensure service reliability, the system starts monitoring processes on certain nodes. The cluster formed by the monitoring processes is called the Paxos control subsystem. The Paxos control subsystem provides node status monitoring and master selection functions. When new nodes are added or nodes become faulty, the Paxos control subsystem will report events to notify the subsystems or modules that pay attention to cluster status changes. As long as the number of faulty nodes in a FusionStorage HDFS for big data cluster does not exceed half of the nodes in the Paxos control subsystem, the cluster can work properly.

The index layer manages metadata as well as stores and reads metadata by interacting with the persistence layer.

The OAM management subsystem is responsible for configuring services and monitoring service and device status. The management subsystem consists of clients and servers. The clients can open the cluster management page through the Internet Explorer. No independent node is provided for the serving end. It is deployed on two nodes that also run the storage subsystem. The two nodes work in active/standby mode. In normal cases, only one node provides services for external requests. If the node fails, the OAM management subsystem switches over to the other node. The switchover is transparent to clients and does not change the IP address of the OAM management subsystem.

Thanks to the distributed architecture, FusionStorage HDFS for big data is able to maintain system availability in the event of any node fault (either man-caused or mechanical). Node overload control further helps minimize the impact of node failures on the whole system.

5.4 Hardware Reliability

Nodes of FusionStorage HDFS for big data leverage the following designs to ensure high reliability:

- Dedicated hot-swappable SAS system disks are used, supporting RAID 0 protection.
- Power and fan modules are redundant.
- Swappable mainboards and cable-free design are adopted, significantly increasing node reliability while reducing 80% of replacement time.
- Triple anti-vibration designs for disks (including using vibration damping screws on fans, enhancing enclosure rigidity, and adding spring washers and damping pads) reduce the disk failure rate and improve reliability of storage nodes.

- The heat dissipation directions of storage nodes are the same. The end-to-end heat dissipation design improves the heat dissipation efficiency, prolongs the service life of electronic components, and ensures stable running of storage nodes in case air conditioners in equipment rooms are abnormal. The cellular porosity rate of front-end panels is 75%. Counter-rotating fans improve wind speed. Flow-dividing air ducts and independent air channels for back-end I/O modules further improve the heat dissipation efficiency.

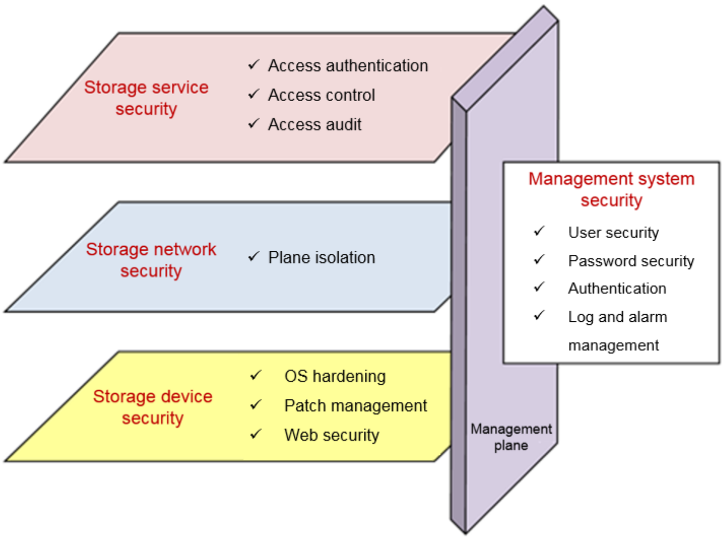
5.5 Link Reliability

Each node uses two ports to interconnect with two stacked service network switches and uses another two ports to interconnect with two stacked storage network switches. The failure of one port or switch will not affect node or system availability. In addition, port bonding implements disaster recovery and makes the most of the port bandwidth.

6 System Security

6.1 Security Architecture

Figure 6-1 Security architecture



6.2 Management System Security

6.2.1 User Security

To prevent mis-operations from compromising storage system stability and service data security, you can define user roles to control user permissions. You can specify user permissions when creating users. Once users have been created, you cannot modify their permissions. After a specified period of idle time, your DeviceManager session automatically times out. In this case, you need to log in again if you still want to access DeviceManager. The session timeout period is modifiable.

Table 6-1 User roles

Role	Permissions
Super administrator	Has full control over the storage device and can create users with different roles.
Administrator	Has the permission to configure and view the Call Home service, alarms, licenses, and SNMP and to view users and user security policies.
System viewer	Has the permission to view users, security policies, alarms, licenses, SNMP, and the Call Home service.
Security administrator	Has the permission to view users, configure system security, and manage security policies, security rules, storage systems, time, certificates, and Key Management CBB (KMC).

6.2.2 Password Security

FusionStorage HDFS for big data supports password complexity policies. A user password must contain at least two of the following character types: special characters, uppercase letters, lowercase letters, and digits. FusionStorage HDFS for big data adopts user login locking mechanisms. The locking mechanism is configurable to prevent brute force cracking. Passwords are encrypted using secure encryption algorithms 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 6-2 Password security policies

Parameter	Description	Value
Min. Length	Minimum length of a password, preventing you from setting overly short passwords.	[Value range] Integer from 8 to 32 [Default value] 8
Max. Length	Maximum length of a password, preventing you from setting overly long passwords.	[Value range] Integer from 8 to 32 [Default value] 16

Parameter	Description	Value
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 Special characters include !"#$%&'()*+,-./:;<=>?@[\]^` {_|}~ and spaces.
Number of Duplicate Characters	Maximum number of times a character can appear consecutively in a password.	[Value range] Integer from 0 to 9 [Default value] 3
Number of Retained Historical Passwords	Number of historical passwords retained for a user. A new password must be different from any of the historical passwords.	[Value range] Integer from 0 to 30. Value **0** indicates no limit. [Default value] 3
Password Validity Period (Days)	Password validity period. You are advised to enable **Password Validity**. After **Password Validity** is enabled, you must set the password validity days. 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)	Password change interval.	[Value range] Integer from 1 to 9999 [Default value] 5

6.2.3 Authentication

FusionStorage HDFS for big data supports IAM, POE, Kerberos, and LDAP authentication, and uses the same Kerberos and LDAP authentication servers as compute clusters.

6.2.4 Log and Alarm Management

- **Log management**

  All operations performed on management planes are recorded in logs. Logs detail event generation time, user IDs (including associated terminals, 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 or deletes logs. Log time is synchronized with a unified time source.

- **Alarm management**

  System exceptions and faults are displayed on DeviceManager in real time, reminding users to handle them. DeviceManager also supports alarm notification by email.

6.3 Storage Service Security

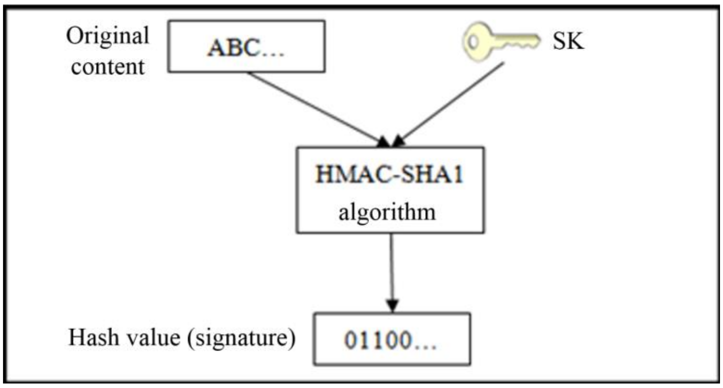
6.3.1 Access Authentication

FusionStorage HDFS for big data employs Kerberos for authentication and LDAP for user management and uses the same Kerberos and LDAP authentication servers as compute clusters.

FusionStorage HDFS for big data uses AKs and SKs to authenticate user identities. During the authentication, keyed-hash message authentication code (HMAC) calculation is performed. 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's AK and a signature calculated based on the user's SK (HMAC uses HMAC-SHA1 and HMAC-SHA256 for signature calculation). After receiving the request, FusionStorage HDFS for big data searches for the SK of the received AK in the system and uses the SK to calculate the signature. Then FusionStorage HDFS for big data compares the calculated signature with that in the user's request. If the two signatures are consistent, the authentication is successful. Figure 6-2 shows signature calculation using an SK.

Figure 6-2 Signature calculation using an SK



6.3.2 Namespace Access Control

FusionStorage HDFS for big data provides flexible and secure access control for data stored in the cloud. You can set access control policies for directories and files in the namespace as required. The access permissions include READ and WRITE. You can also grant other users permissions to access and set control policies for your directories and files.

6.3.3 Access Audit

FusionStorage HDFS for big data records all non-query user activities and background operation instructions in logs. The logs can be used for auditing and operation tracing.

6.4 Storage Network Security

6.4.1 Plane Isolation

FusionStorage HDFS for big data consists of service plane, storage plane, management plane, BMC plane, and control plane. All the planes are isolated from each other to ensure system security. For details, see section 3.5 System Networking.

6.5 Storage Device Security

6.5.1 Operating System Hardening

FusionStorage HDFS for big data uses EulerOS. The following measures are taken to protect the operating system:

- **Operating system tailoring**

  Unnecessary services and components are deleted or disabled to reduce the size of the operating system. This improves the startup speed and security of the operating system without affecting support for desired services and existing features.
- **System service security hardening**

Insecure services, such as Telnet, SNMPv1, SNMPv2c, and FTP, as well as unnecessary or risky background processes and services are disabled. Secure communication and transmission protocols are adopted. For example, SSH v2 is used instead of Telnet.

- **Kernel security hardening**

  Execution stacks are protected against buffer overflow attacks. Functions, such as IP address forwarding, response to broadcast requests, and Internet Control Message Protocol (ICMP) redirects receiving, are disabled. TCP-SYN cookie protection is enabled to prevent SYN attacks (DoS attacks).

- **Account and password protection**

  Unnecessary users and user groups have been deleted. The password complexity check function has been enabled, and password validity periods and the number of login attempts have been configured.

- **File and directory permission control**

  Permissions on files and directories are minimized in accordance with security hardening specifications and application requirements in the industry.

- **Logs and audit**

  Logs of service running and kernel processes are recorded.

6.5.2 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. Huawei adopts the following security patch management measures to enhance system security:

- Periodically provides on-demand security patches for users as operating system security patches and open-source software security patches are released.
- Releases patches to fix security vulnerabilities in Huawei FusionStorage HDFS for big data based on vulnerability severities.

6.5.3 Web Security

DeviceManager, the management GUI of FusionStorage HDFS for big data, provides the following web security enhancements:

- **Secure access over HTTPS**

  DeviceManager only supports secure access channels over HTTPS, enhancing access security.

- **Prevention against cross-site scripting (XSS) attacks**

  XSS attacks occur when attackers use a vulnerable web application to send malicious code to users.

- **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.

- **Prevention against cross-site request forgery (XSRF)**

  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.

- **Protection for sensitive information**

  DeviceManager hides sensitive information to prevent interception by attackers.

- **Restriction on file upload and download**

  DeviceManager restricts file upload and download 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.

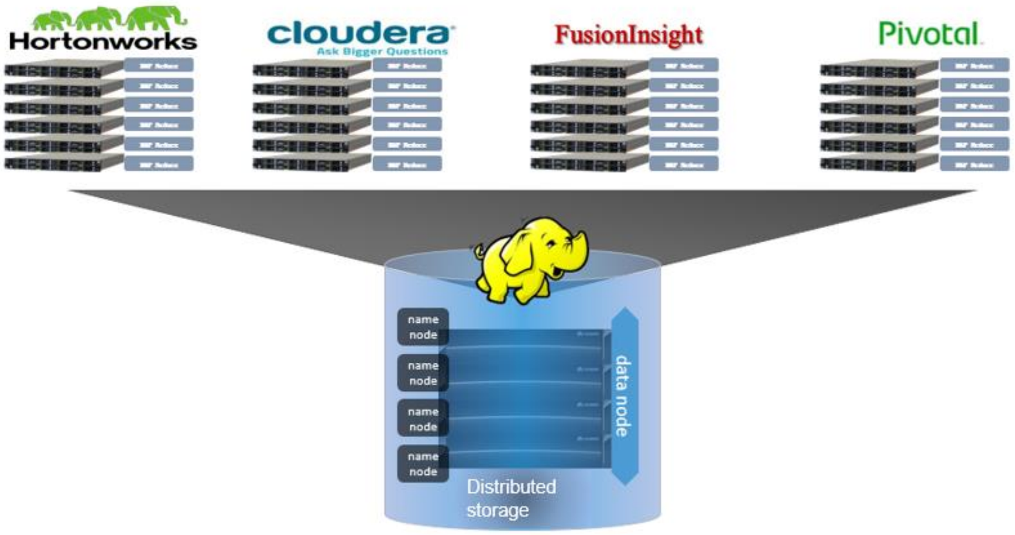
7 Openness and Compatibility

7.1 Mainstream Protocols

Huawei FusionStorage HDFS for big data is fully compatible with native HDFS semantics.

7.2 Big Data Platforms

FusionStorage HDFS for big data uses Huawei-developed HDFS to provide complete HDFS semantics. In such big data storage architecture, FusionStorage HDFS for big data is extensively compatible with Huawei and third-party big data platforms.



Compared with S3A, FusionStorage HDFS for big data has the following advantages in simulating file system semantics on object storage:

- The I/O stack is directly connected to the metadata layer and data layer, avoiding extra overhead caused by protocol conversion.
- In terms of protocol compliance, FusionStorage HDFS for big data supports complete directory tree semantics such as renaming and moving, preventing extra data copy costs and eliminating huge performance loss caused by object storage such as S3A simulating semantics. In addition, FusionStorage HDFS for big data is fully compatible with Hadoop security authentication and is better compatible with the Hadoop/Spark upper-layer application ecosystem.

FusionStorage HDFS for big data supports complete HDFS semantics and provides the functions that are not supported by S3A. The following table compares the main functions of FusionStorage HDFS for big data with S3A.

Table 7-1 Main functions of FusionStorage HDFS for big data and S3A

Operation	FusionStorage HDFS for Big Data	S3A	Disadvantage of S3A
Rename	Renames a file or directory. Computing frameworks such as MapReduce/Spark export data to a **.tmp** file first, and then rename the file to generate a finalized formal file.	Copy + Delete	Copying requires extra cache space and results in large system overhead and low performance. Data consistency cannot be guaranteed.
Move	Moves a file or a directory to another parent directory.	Copy + Delete	
List Directory	Lists all contents in a directory.	Scans and filters all objects whose name prefixes contain the directory name.	Sub-directories and files are also scanned and filtered, resulting in low performance.
Delete Directory	Deletes a directory and all contents in the directory.	Scans all objects whose name prefixes contain the directory name and deletes them one by one or in a batch.	Deletion takes a long time, and data consistency cannot be guaranteed.

Operation	FusionStorage HDFS for Big Data	S3A	Disadvantage of S3A
Append	Appends data to a closed file.	Not supported	The application layer can simulate this operation by creating new objects. However, too many small objects may be generated.
Flush	Ensures that data is flushed to the storage server.	Not supported	Data security cannot be ensured before objects are written to disks. Data cannot be recovered to a specific point in time.
Sync	Ensures that data is stored on disks.	Not supported	
Truncate	Truncates files from a certain location.	Not supported	
ACK	Implements error tolerance if a fault, such as network timeout, occurs during a write operation.	Not supported	If a network error occurs when a large file is being written, data written in the file cannot be restored
Authenticate	Has a unified authentication with compute clusters.	Big data authentication is not supported.	Security authentication is not supported.

7.3 Centralized 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.

IT O&M management platform software generally accesses IT infrastructure using the SNMP protocol. Huawei FusionStorage HDFS for big data complies with SNMP and REST protocols and provides open APIs to support different features.

8 Acronyms and Abbreviations

CLI	command-line interface
CMS	cluster management service
DAS	direct-attached storage
DHT	distributed hash table
DNS	Domain Name System
DS	data service
EC	erasure coding
GID	group ID
GUI	graphical user interface
IPMI	Intelligent Platform Management Interface
NS	namespace
Paxos	A highly fault-tolerant consistency algorithm based on message transference
RAID	redundant array of independent disks
RDMA	remote direct memory access
SAS	serial-attached SCSI
SATA	serial advanced technology attachment
SSD	solid-state drive
TCP	transmission control protocol
UID	user identity