Huawei Certification Cloud Computing Training Courses

HCIA-Cloud Computing Learning Guide

Version: V4.0

HUAWEI TECHNOLOGIES CO., LTD.

Copyright © Huawei Technologies Co., Ltd. 2019 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://www.huawei.com/

Email: support@huawei.com

Contents

1 What's Cloud Computing? 1
1.1 Cloud Computing Is Around Us 1
1.2 Characteristics of Cloud Computing 3
1.2.1 On-Demand Self-Service 3
1.2.2 Broad Network Access 4
1.2.3 Resource Pooling 5
1.2.4 Rapid Elasticity 6
1.2.5 Measured Service 6
1.3 Definition of Cloud Computing 7
1.4 Origin and Development of Cloud Computing 8
1.4.1 A Brief History of the Internet 8
1.4.2 The History of Computing 10
1.4.3 Development of Cloud Computing 13
1.4.4 Further Reading: Differences Between Cloud Computing 1.0 and 2.0/3.0 15
1.5 Cloud Computing Models 20
1.5.1 By Deployment Model 21
1.5.2 By Service Model 22
2 Introduction to Compute Virtualization 24
2.1 Virtualization Overview 24
2.1.1 What's Virtualization? 24
2.1.2 A Brief History of Compute Virtualization 26
2.1.3 Compute Virtualization Types 27
2.2 Compute Virtualization 29
2.2.1 CPU Virtualization 29
2.2.2 Memory Virtualization 35
2.2.3 I/O Virtualization 36
2.2.4 Mainstream Compute Virtualization 37
2.3 KVM 38
2.4 FusionCompute 42
3 Network Basics for Cloud Computing 44
3.1 Network Architecture in Virtualization 44
3.1.1 Traffic on a Virtual Network 44

3.1.2 Basic Network Concepts 45
3.2 Physical Networks in Virtualization 49
3.3 Virtual Networks in Virtualization 54
3.3.1 Virtual Network Architecture 54
3.3.2 Network Features of Huawei Virtualization Products 61
4 Storage Basics for Cloud Computing 69
4.1 Mainstream Physical Disk Types 70
4.1.1 HDD 70
4.1.2 SSD 74
4.2 Centralized Storage and Distributed Storage 77
4.2.1 Centralized Storage 77
4.2.2 RAID 85
4.2.3 Distributed Storage and Replication 88
4.3 Virtualized Storage and Non-virtualized Storage 92
4.4 VM Disks 95
4.5 Storage Features of Huawei Virtualization Products 95
4.5.1 Storage Architecture of Huawei Virtualization Products 95
4.5.2 Characteristics of Huawei VM Disks 96
5 Virtualization Features 98
5.1 Virtualization Cluster Features 98
5.1.1 HA 99
5.1.2 Load Balancing 100
5.1.3 Scalability 101
5.1.4 Memory Overcommitment 102
5.2 VM Features 104
5.2.1 Quick VM Deployment 104
5.2.2 VM Resource Hot-Add 105
5.2.3 VM Console 106
5.2.4 VM Snapshot 107
5.2.5 NUMA 108
5.3 Huawei Virtualization Product Features 109
5.3.1 Cluster Features 109
5.3.2 VM Features 112
6 Cloud Computing Trends 115
6.1 Fields Related to Cloud Computing 115
6.2 Cloud-Enabling Technologies 121
6.2.1 Container 121
6.2.2 OpenStack 123
6.3 Other Emerging Technologies 129
6.3.1 Fog Computing 129
6.3.2 Edge Computing 129

6.3.3 Microservices 130
6.3.4 Serverless 131
7 Conclusion 132
8 Appendix 134
8.1 Verification 1 134
8.2 Verification 2 135
8.3 Verification 3 136
8.4 Verification 4 137

1 What's Cloud Computing?

1.1 Cloud Computing Is Around Us

People outside of the tech industry may have many questions when they so often come across the term cloud computing. What's cloud computing? What kind of services does cloud computing provide? Where and how do I acquire them?

Cloud computing may be a technical term whose meaning is unclear to many, but there is a good chance that many of us are already using cloud services without being aware it.

First, let's take a look at the **Products** menu on the HUAWEI CLOUD user portal, Huawei's public cloud service, as shown in Figure 1-1.

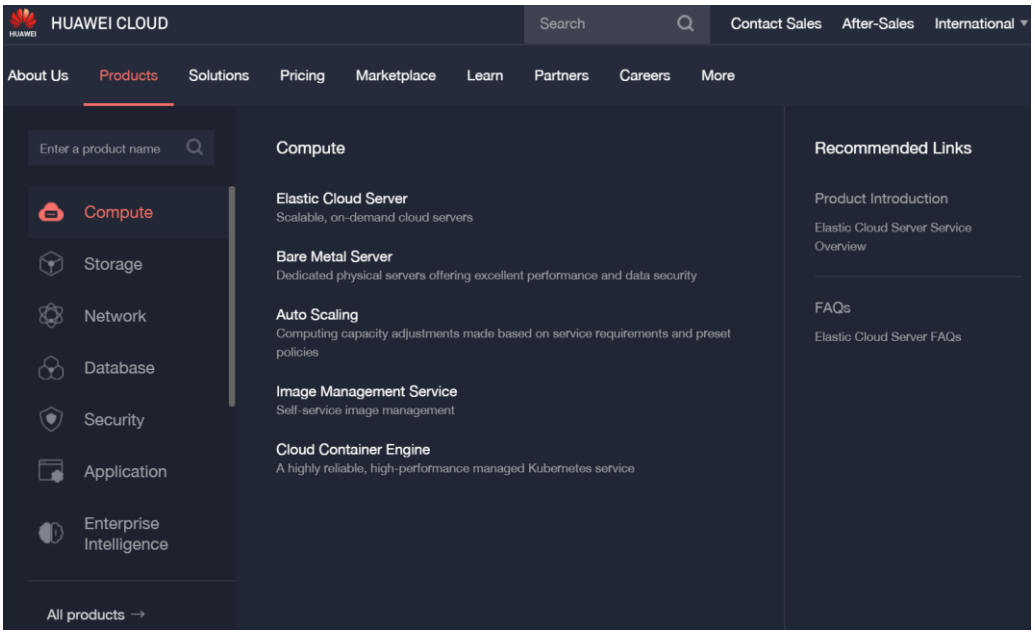


Figure 1-1 HUAWEI CLOUD user portal

Under **Products**, we can see several service categories, including Compute, Storage, Network, Application, and more. Each category further contains a variable number of services. Now, let's have a look at one of the most popular cloud services on HUAWEI CLOUD — Elastic Cloud Server (ECS). Figure 1-2 shows available ECS flavors (or specifications).

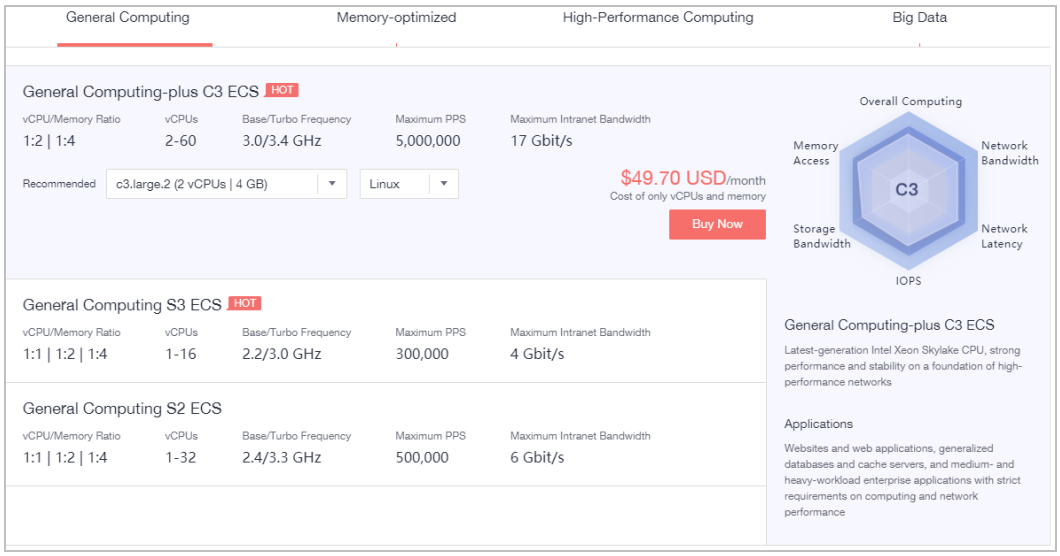


Figure 1-2 ECS flavors

An ECS flavor is similar to the computer hardware specifications we see when we buy a new computer. They both contain CPU, memory, hard disk, and other parameters. To ready an ECS, we will also need to install the required OS and configure an IP address for it. In fact, subscribing to an ECS is just like buying and setting up a personal computer, except that an ECS is a computer on the cloud. It can do almost anything that a conventional computer does, such as editing documents, sending emails, and enabling office collaboration, plus the things that a conventional computer can't do. For example, an ECS is accessible via a mobile phone or tablet, with a similar user experience as when it is accessed through a computer with a big screen monitor. Also, you can modify the configuration of your ECS at any time, for example, scaling the ECS memory from 1 GB to 2 GB.

In addition to ECS, you can also subscribe to many other services on HUAWEI CLOUD. For example, you can buy the CloudSite service to quickly build websites, or the Object Storage Service (OSS) or Elastic Volume Service (EVS) to quickly expand storage spaces. HUAWEI CLOUD, as well as other mainstream cloud platforms, also provide AI services, such as facial, voice, image, and text recognition.

In short, cloud computing allows us to use IT services as conveniently as using utilities like water and electricity. Think about how we use water and electricity. We use water and electricity simply by turning on the tap or power switch. This is because water and electricity are already on the grids. This is also true with IT services. Cloud computing delivers ready-to-use IT services over the Internet. By analogy, the Internet is the grid, and web portals or apps are the water taps

You may say: I have my own personal computer. Most applications I need are installed on my local disk, and I don't use services like facial or voice recognition. IT services are not water or electricity. I can live without them. So cloud computing has nothing to do with me. There is a chance that you might be wrong, for you may be using cloud computing already.

As we have said earlier, the "tap" for cloud computing, that is, how and where we access cloud resources, may be a web page or app. Here are a few apps you might be using already: auto backup & restore on Huawei phones, Google Translate, and iReader.

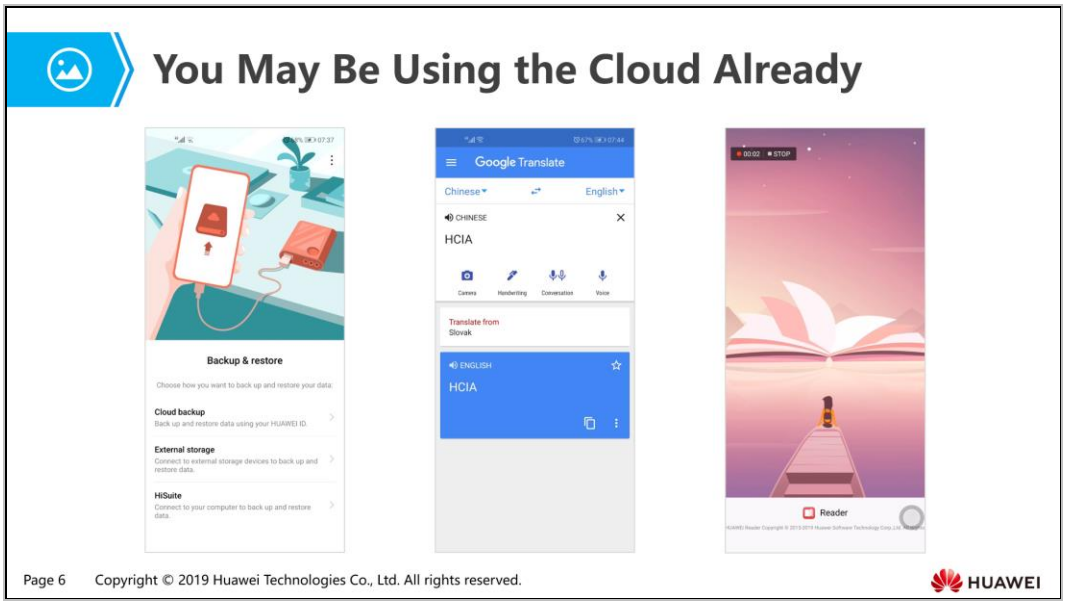


Figure 1-3 Cloud computing apps

Backup & Restore is a default service on Huawei phones. Other brands also have similar services, such as iCloud for iPhone. These services allow you to back up the local files on your phone to a remote data center. After you change to a new phone, you can easily restore your data to your new phone using your account and password configured for this service.

Google Translate is a free service that instantly translates words, phrases, and web pages between English and over 100 other languages.

iReader is a popular online reading app that gives you access to a huge library of online electronic books.

These three apps are all powered by the cloud. Even if you never used any of them, there is a good chance you are using other cloud-based apps without being aware of it.

1.2 Characteristics of Cloud Computing

As cloud computing services mature both commercially and technologically, it will be easier for companies as well as individuals to reap the benefits of cloud. Cloud computing has the following five well-recognized characteristics.

1.2.1 On-Demand Self-Service

Supermarket is a good example of on-demand self-service. In a supermarket, each consumer chooses the items they need by themselves. For similar goods of the same category, we make our choice by comparing their prices, brands, and product descriptions. On-demand self-service is also one of the major characteristics of cloud computing. A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service's provider.

The prerequisite for on-demand self-service is for the consumer to understand their own needs and know which product or products can accommodate such needs. A supermarket is comparable to a cloud service platform in the sense that they both offer a huge variety of products (called cloud services in the case of the cloud platform). To deliver a good consumer

experience, the supermarket or cloud provider may need to make necessary recommendations or consumer guidance in order to help consumers make the right choices.

1.2.2 Broad Network Access

Cloud computing is computing power over the Internet, so network access is an innate characteristic of cloud computing.

Today, the Internet has reached almost every inhabited corner of the world. We can use any electronic device, such as a personal computer, tablet, or cell phone to connect to the Internet. This means we can have access to cloud services through any electronic device as long as there is network connectivity. When in office, we can use cloud services via personal computers. In air ports or train stations, we can use them through a mobile phone or tablet over Wi-Fi or mobile data.

Figure 1-4 and Figure 1-5 both show the user interface for ModelArts, a one-stop development platform for AI developers offered by HUAWEI CLOUD: one on a personal computer and the other on a mobile phone. Despite the different user interfaces, the experiences are the same.

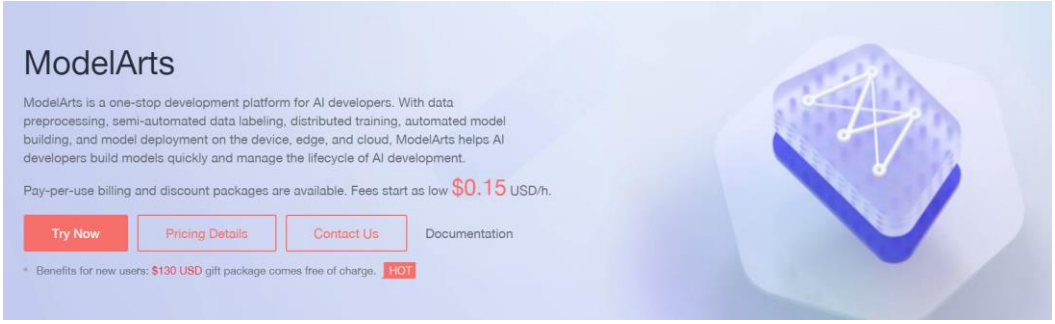


Figure 1-4 ModelArts on a personal computer

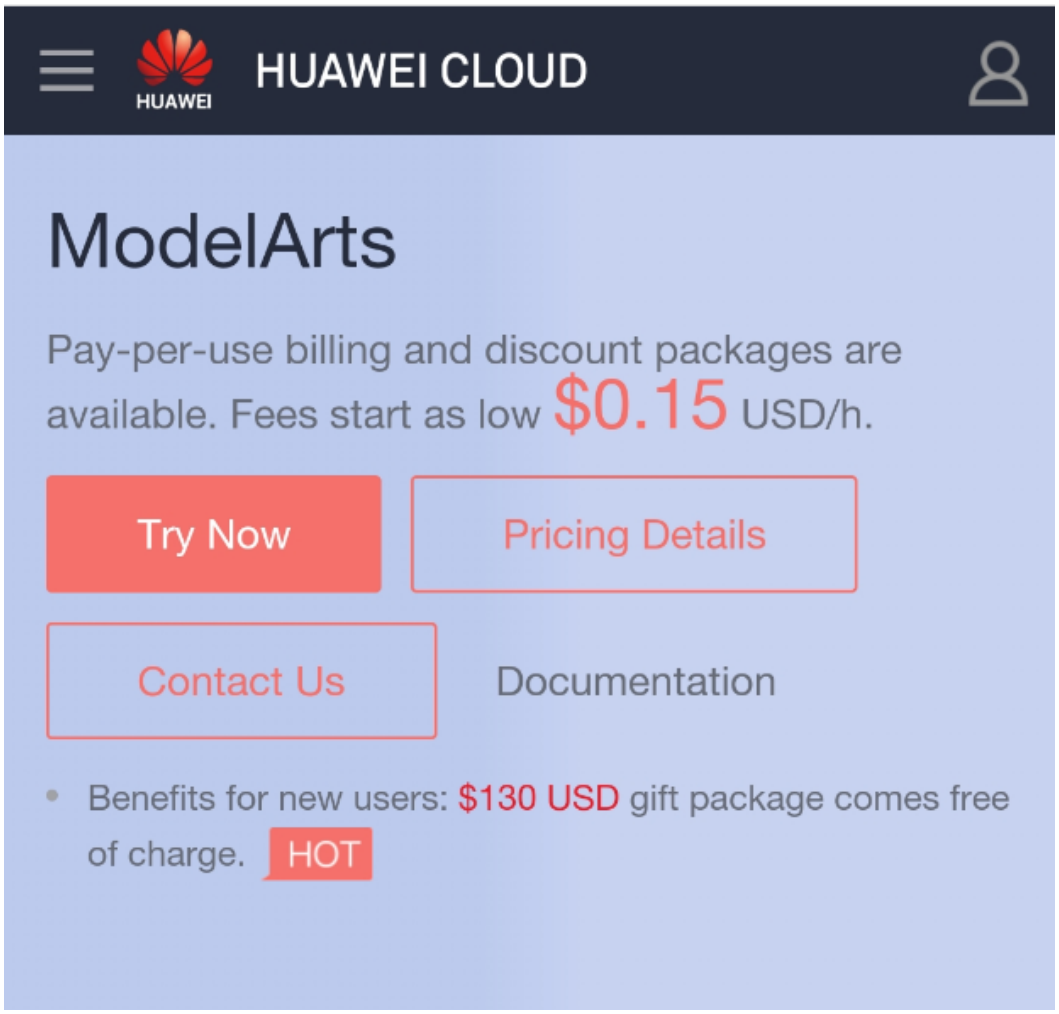


Figure 1-5 ModelArts on a mobile phone

1.2.3 Resource Pooling

Resource pooling is one of the prerequisites for on-demand self-service. In a supermarket, we can see that different categories of items are put into different areas, such as the fruit and vegetable area, the fast frozen food area, and so on, so that consumers can quickly find the items they need.

Resource pooling is not merely putting all resources of the same type onto the same rack, as it is done in supermarkets. Resource pooling is also about breaking down the resources by the finest granularities for flexible, on-demand provisioning.

Instant noodles are popular food among many. The problem is that, for some people, one bag is too little but two are too much. In theory, the manufacturers can solve this problem by reducing the minimum purchasable unit for instant noodles from per bag to an even smaller one. This is one of the things resource pooling does. Another example is how drinks are served in many cafeterias. Different types of fruit juices are put into different dispensers so that customers can take the exact amounts they need.

Another thing that resource pooling does is to shield the differences in the underlying resources. Again we use the analogy of serving carbonated drinks in cafeterias. Customers may not know whether they are having Pepsi or Coca-Cola, because the dispenser says nothing of the brand. Resources that can be pooled include compute, storage, and network

resources. Compute resources include CPU and memory. Pooled CPU resources are available to consumers per core. Consumers have no idea whether they are use AMD or Intel CPUs. Further details about pooled compute resources will be provided in section "Computing Virtualization" of this book.

1.2.4 Rapid Elasticity

For applications that may face fluctuations in demand, being able to rapidly and elastically provision computing resources is a desirable feature. When the load surges during peak hours or in the case of a major social or commercial event, more servers can be quickly provisioned, and automatically in some cases. When the load reduces, the servers can be quickly released for reallocation.

Rapid elasticity, both scaling in or out, can be achieved manually or based on predefined policies. Scaling can be done by increasing or decreasing the quantity of servers, or by increasing or decreasing the resources available with each server.

A best example of this characteristic, which the Chinese are familiar with, is the primary weapon of the Monkey King (or Sun Wukong), called Ruyi Jingu Bang. This weapon is a rod which he can shrink down to the size of a needle and keep in his ear, as well as expand it to gigantic proportions, as need be. Besides, he can also pluck hairs from his body and blow on them to convert them into clones of himself to gain a numerical advantage in battle. In this case, the rod also multiplies so that each of the clones can have one.

The most significant benefit of rapid elasticity is cost reduction with guaranteed business continuity and reliability. For example, with this characteristic, a startup company can start by acquiring only small amounts of IT resources and add more as its business grows. The company can quickly obtain more resources to handle load surges and release them afterwards. This allows the company to spend its limited budget on higher-priority aspects of their business.

1.2.5 Measured Service

Measured service is how cloud systems control a user or tenant's use of resources by leveraging a metering capability. Metering is not billing, although billing is based on metering.

In cloud computing, most services came with a price while some others are free. For example, Auto Scaling can be provisioned as a service, and most of the times this service is free of charge.

Measured service ensures that all resource usage can be accurately measured, through technical or other means, based on predefined criteria, which can be the duration of usage, resource quota, or the volume of data transmitted. With these, cloud systems can automatically control and adjust resource configuration based on usage.

On the cloud consumers' part, they can know the exact usage of the services they have subscribed, and decide whether to scale up or down based on the current usage.

Let's again resort to the example of the Monkey King's rod. The Monkey King can change the size of the rod freely to suit his needs, which are mostly to combat demons of various kinds and shapes. For example, facing the Skeleton Queen, he can turn the rod to 3 meters long and maybe 10 cm thick and keep it at this state for 30 minutes. For a less powerful demon, he may turn the rod to a smaller size, for example, 2 meters long and 7 cm thick, and keep it this way for 15 minutes. When at rest, he can turn the rod to 1 cm long and 0.1 cm thick and so to keep it in his ear.

Copyright © Huawei Technologies Co., Ltd.

This shows how cloud service instances can be scaled based on accurate measurement of service usage.

Figure 1-6 shows an example of measured usage of the ECS service.

For example, for an ECS, the result can be 1 vCPU, 2 GB memory, and 40 GB disk space, with a validity period of one month.

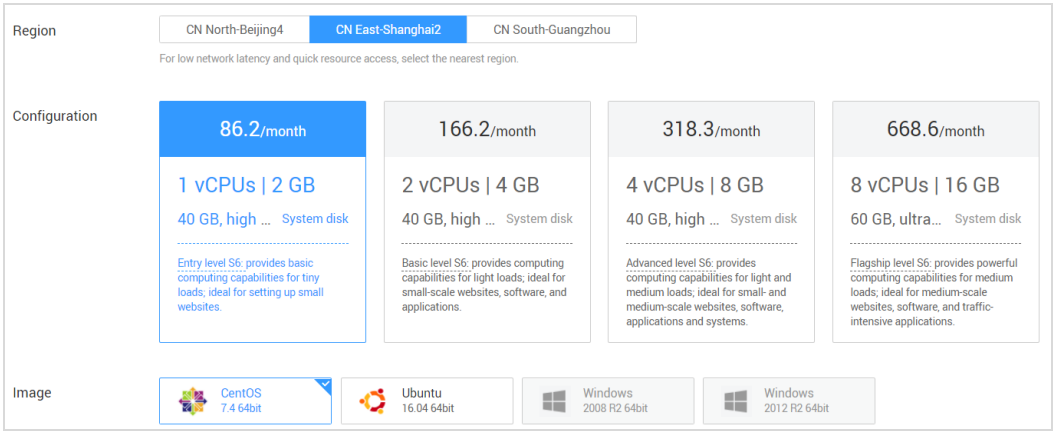


Figure 1-6 Measured ECS service

1.3 Definition of Cloud Computing

The IT industry must have at least one hundred definitions of what cloud computing is. One of the most widely accepted is given by the National Institute of Standards and Technology (NIST) of the US: A model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Note the following key points in this definition:

- Cloud computing is not a technology so much as a service delivery model.
- Cloud computing gives users convenient access to IT services, including networks, servers, storage, applications, and services, like using utilities such as water and electricity.
- The prerequisite for convenient, on-demand access to cloud resources is network connectivity.
- Rapid resource provisioning and reclamation fall into the rapid elasticity characteristic of cloud computing, while minimal management effort and service provider interaction the on-demand self-service characteristic.

In the term "cloud computing", "cloud" is a metaphor for the Internet. It is an abstraction of the Internet and the infrastructure underpinning it. "Computing" refers to computing services provided by a sufficiently powerful computer capable of providing a range of functionalities, resources, and storage. Put together, cloud computing can be understood as the delivery of on-demand, measured computing services over the Internet.

NOTE

The word "cloud" comes from the cloud symbol used by datacom engineers in flow charts and diagrams to symbolize the Internet or other types of networks.

The history of cloud computing consists of those of the Internet and computing models. In the next chapter, we will talk about how cloud computing develops into what it is today. But before that, let's hear a story about a failed attempt to popularize cloud computing before the Internet reaches its maturity.

This story is about Larry Ellison, co-founder and the executive chairman and chief technology officer of Oracle Corporation, and a legend in the IT industry. If you're interested, you can search for him on the Internet. Around the time Oracle was founded, two other legendary US companies, Apple and Microsoft were also founded. To the general public, Larry Ellison always seemed to be a bit overshadowed by Bill Gates. At the beginning, Microsoft's business was computer operating systems, and Oracle's was databases. However, in 1988, Microsoft also launched the SQL Server database. This seemed a direct challenge to Oracle. In response, Larry Ellison launched an Internet computer without an OS or hard disk. Rather, the OS, user data, and computer programs all resided on servers located in a remote data center. This product also had a price advantage over computers running Microsoft OSs. However, there was a small miscalculation in Larry Ellison's plan: The year was 1995, and the Internet was still at its infancy. At that time, the Internet was still unavailable in most parts of the world, and was unable to provide the bandwidth needed for the Internet computer to function properly. This led to poor user experience, so the project was terminated two years later.

The Internet computer launched by Oracle can be seen as an early form of cloud computing. The only problem was that it was way ahead of its time. Plus, the Doc-Com bubble burst around 2000 also affected people's confidence in cloud-based applications. This situation lasted until Amazon launched AWS in 2006.

1.4 Origin and Development of Cloud Computing

By definition, cloud computing can be understood as the delivery of on-demand, measured computing services over the Internet. The history of cloud computing consists of those of the Internet and computing models. This chapter will talk about the histories of all three.

1.4.1 A Brief History of the Internet

In the beginning, all computers were separated from each other. Data computation and transmission were all done locally. Later, the Internet was born connecting all these computers, and also the world together. The following are some of the milestone events in the history of the modern Internet.

1969: The Advanced Research Projects Agency Network (ARPANET) was born, and is widely recognized as the predecessor of today's Internet. Like many technologies that are underpinning our modern society, the ARPANET was originally developed to serve military purposes. It is said that the ARPANET was launched by the US military to keep a fault-tolerant communications network active in the US in the event of a nuclear attack. In the beginning, only four nodes joined the ARPANET. They were four universities in the central states of the US: the University of California, Los Angeles (UCLA), Stanford Research Institute (SRI), University of California, Santa Barbara (UC Santa Barbara), and University of Utah. The birth of the ARPANET marked the beginning of the Internet era. In the coming years, more and more nodes joined the ARPANET, and the majority of them came from non-military fields. In 1983, for security reasons, 45 nodes were removed from ARPANET to form a separate military network called MILNET. The remaining nodes were used for civilian purposes.

1981: The complete specifications of the TCP/IP protocol suite were released for the first time, signaling the birth of the Internet communications language. Why was the TCP/IP protocol

needed? TCP/IP is in fact a suite of many protocols, including the Transport Control Protocol (TCP), Internet Protocol (IP), and others. The earliest protocol used on the ARPANET was called the Network Control Protocol (NCP). However, as the ARPANET grew, NCP could not keep up with the demands of large-scale networks. Born for use on large and mega-size networks, TCP/IP was soon used by the ARPANET to replace NCP on January 01, 1983.

1983: All three of the original networks, ARPANET, PRNET, and SATNET, switched to TCP/IP, which marked the beginning of the accelerated growth of the Internet.

1984: The Domain Name System (DNS) was invented. Since the adoption of TCP/IP, the development of the Internet picked up speed, and more computers were added to the network. Each computer used TCP/IP-compliant numerical IP addresses to identify each other. As the quantity of connected computers continued to increase, the inconvenience of using IP addresses to identify computers became evident: they are hard to memorize. This is comparable to using people's identity numbers, instead of their names, to identify them. It's difficult to memorize such long numbers. This is where DNS came in. DNS translates between numerical IP addresses and more readily memorized domain names. In this way, computer users can locate their peers simply through domain names, leaving the translation work to domain name servers. The domain name consists of two parts: name, for example, **HUAWEI;** and category or purpose, for example, **.com** for commercial. Maintenance personnel can enter the domain name **HUAWEI.com** to reach the computer with the corresponding IP address. Today, domain names, used in URLs, can be used to identify any web pages across the globe.

1986: The modern email routing system MERS was developed.

1989: The first commercial network operator PSINet was founded. Before PSINet, most networks were funded by the government or military for military or industrial purposes, or for scientific research. PSINet marked the beginning of the commercial Internet.

1990: The first network search engine Archie was launched. As the Internet expanded, the amounts of information on the Internet grew at an explosive rate. A search engine or website was needed to index and search for information needed by users, to speed up the searches. Archie is the earliest search engine and a tool for indexing FTP archives located on physically dispersed FTP servers. It was developed by Alan Emtage, then a student at McGill University. Archie allows users to search for files by their names.

1991: WWW was officially open to the public. The World Wide Web (WWW), or simply the Web, that most of us now use on a daily basis, became publicly available only in 1991, less than 30 years ago. Tim Berners-Lee, a British scientist, invented the Web while working at CERN, the European Organization for Nuclear Research. The Web allows hypermedia, which can be documents, voice or video, and a lot more, to be transmitted over the Internet. It was only after the popularization of the Web that all the great Internet companies were born and all kinds of Internet applications that have fundamentally changed the lives of ordinary people began to emerge.

1995: E-commerce platforms Amazon and eBay were founded. Many great companies, such as Yahoo and Google, emerged since the brief history of the Internet began. Here we will talk about Amazon alone, since it's the first Internet company that made commercial cloud computing a reality. In the early days, Amazon mainly sold books online. To process and store commodity and user information, Amazon built huge data centers. The US has a shopping festival called Black Friday, similar to the "Double Eleven" invented by Tmall of China. On this day, Amazon needed to process huge amounts of information, and all the servers in its data centers were used. However, after this day, most of the servers were idle. To improve return on investment, Amazon needed to lease these idle servers. This was the reason why in 2006 Amazon launched its first cloud computing product: Elastic Compute Cloud (ECS).

Copyright © Huawei Technologies Co., Ltd.

NOTE

Many of the best known companies, such as Amazon, Google, Alibaba, and Tencent, are Internet companies. Companies like IBM, Cisco, Huawei, and Lenovo are traditional IT companies.

2000: The dotcom bubble burst. The unprecedented growth of the Internet in the 1990s resulted in the dot-com bubble, which burst around 2000. It was during this period that PSINet, the first commercial network operator we mentioned earlier, went bankrupt.

The Internet regained rapid growth after the dotcom bubble burst. In 2004, Facebook was founded, and with it came the phenomenon of social networking.

1.4.2 The History of Computing

1.4.2.1 Parallel computing

Traditionally, software has been written for serial computation:

1. Each problem is broken into a discrete series of instructions.
2. Instructions are executed one after another on a single CPU.
3. Only one instruction may execute at any point in time.

Figure 1-7 Schematic diagram of serial computing

With serial computing, a complex problem takes a long time to process. With large-scale applications, especially when there is limited computer memory capacity, a single-CPU architecture is impractical or even impossible. For example, a search engine and networked database process millions of requests per second, which is far beyond the capacity of serial computing.

Limits to serial computing, both in theory and for practical reasons, pose significant constraints to simply building ever faster serial computers:

- Transmission speeds — the speed of a serial computer is directly dependent upon how fast data can move through hardware. Absolute limits are the speed of light (30 cm/nanosecond) and the transmission limit of copper wire (9 cm/nanosecond). Increasing speeds necessitate increasing proximity of processing elements.
- Limits to miniaturization — processor technology is allowing an increasing number of transistors to be placed on a chip. However, even with molecular or atomic-level components, a limit will be reached on how small components can be.
- Economic limitations — it is increasingly expensive to make a single processor faster. Using a larger number of moderately fast commodity processors to achieve the same (or better) performance is less expensive.

In the simplest sense, parallel computing is the simultaneous use of multiple compute resources to solve a computational problem.

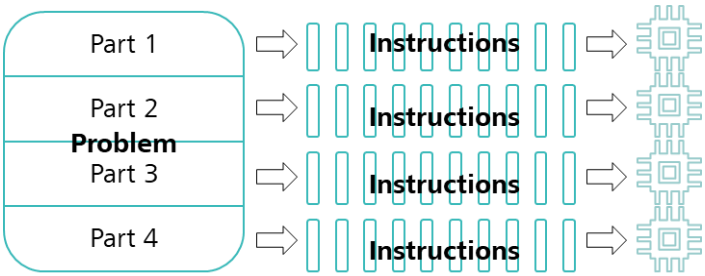


Figure 1-8 Schematic diagram of parallel computing

- Each problem is broken into discrete parts that can be solved concurrently.
- Each part is further broken down to a series of instructions.
- Instructions from each part execute simultaneously on different CPUs.
- A unified control mechanism is added to control the entire process.

Traditionally, parallel computing has been considered to be "the high end of computing" and has been used for cases such as scientific computing and numerical simulations of complex systems. Today, commercial applications are providing an equal or greater driving force in the development of faster computers. These applications require the processing of large amounts of data in sophisticated ways.

The reasons for using parallel computing include the following:

- Time and cost savings: In theory, using more compute resources will lead to completing a task faster and save potential costs. This is even more true considering the resources can be inexpensive, and even out-of-date CPUs clustered together.
- Solving larger problems that can be handled using serial computing.

The CPUs used for parallel computing can come from the same computer, or from different computers residing on the same network.

1.4.2.2 Distributed Computing

Distributed computing is a field of computer science that studies distributed systems. A distributed system distributes its different components to different networked computers, which communicate and coordinate their actions using a unified messaging mechanism. The components work together in order to achieve a common goal.

Distributed computing provides the following benefits:

- Easier resource sharing
- Balanced load across multiple computers
- Running each program on the most eligible computers

The first two are the core rationale behind distributed computing.

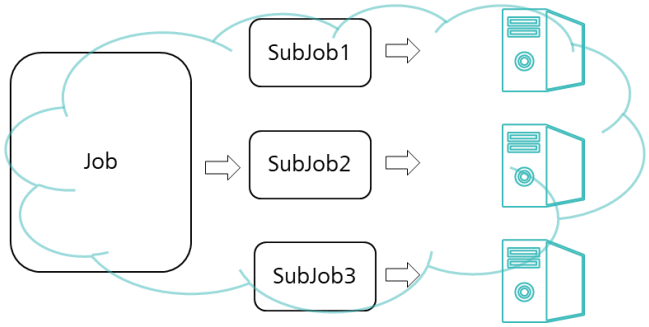


Figure 1-9 Schematic diagram of distributed computing

Parallel and distributed computing both use parallelism to achieve higher performance. Their difference lies in how memory is used: In parallel computing, the computer can have a shared memory or distributed memory. In distributed computing, each computer has its own memory. Some people believe that distributed computing is a special case of parallel computing.

In fact, in distributed computing, each task is independent. The result of one task, whether unavailable or invalid, virtually does not affect other tasks. Therefore, distributed computing has low requirement on real-timeliness and can tolerate errors. (Each problem is divided into many tasks, each of which is solved by one or more computers. The uploaded results are compared and verified in the case of a large discrepancy.)

In parallel computing, there are no redundant tasks. The results of all tasks affect one another. This requires the correct result to be obtained for each task, and preferably in a synchronized manner. In the case of distributed computing, many tasks are redundant, and many useless data blocks are generated. Despite its advantage in speed, the actual efficiency may be low.

1.4.2.3 Grid Computing

Grid computing is the use of widely distributed computer resources to reach a common goal. It is a special type of distributed computing. According to IBM's definition, a grid aggregates compute resources dispersed across the local network or even the Internet, making the end users (or client applications) believe that they have access to a super and virtual computer. The vision of grid computing is to create a collection of virtual and dynamic resources so that individuals and organizations can have secure and coordinated access to these resources. Grid computing is usually implemented in the form of a cluster of networked, loosely coupled computers.

1.4.2.4 Cloud Computing

Cloud computing is a new way of infrastructure sharing. It pools massive amounts of resources together to support a large variety of IT services. Many factors drive up the demand for such environments, including connected devices, real-time stream processing, adoption of service-oriented architecture (SOA), and rapid growth of Web 2.0 applications such as search, open collaboration, social network, and mobile office. In addition, improved performance of digital components has allowed even larger IT environments to be deployed, which also drives up the demand for unified clouds. Cloud computing is hailed as a revolutionary computing model by its advocates, as it allows the sharing of super computational power over the Internet. Enterprises and individual users no longer need to spend a fortune purchasing expensive hardware. Instead, they purchase on-demand computing power provisioned over the Internet.

Cloud computing in a narrow sense refers to a way of delivering and using IT infrastructure to enable access to on-demand and scalable resources (infrastructure, platform, software, etc.) The network over which the resources are provisioned is referred to as the cloud. To

consumers, the resources on the cloud appear to be infinite. They are available and scalable on demand and use a pay-as-you-go (PAYG) billing model. These characteristics allow us to use IT services as conveniently as using utilities like water and electricity.

In a broad sense, cloud computing refers to the on-demand delivery and utilization of scalable services over the Internet. These services can be IT, software, Internet, or any other services.

Cloud computing has the following typical characteristics:

- Hyperscale. Clouds are usually large. Google's cloud consists of over 1 million servers. The clouds of Amazon, IBM, Microsoft, and Yahoo each has hundreds of thousands of servers. The private cloud of an enterprise can have hundreds to thousands of servers. The cloud offers users computational power that was impossible with conventional methods.
- Virtualization. Cloud computing gives users access to applications and services regardless of their location or what device they use. The requested resources are provided by the cloud, rather than any tangible entity. Applications run somewhere in the cloud. The users have no knowledge and do not need to concern themselves about the locations of the applications. A laptop or mobile phone is all they need to access the services they need, or even to perform complex tasks like supercomputing.
- High reliability. With mechanisms such as multi-copy redundancy, fault tolerance, fast and automated failover between homogeneous compute nodes, it is possible for cloud computing to deliver higher reliability than local computers.
- General-purpose. A single cloud is able to run a huge variety of workloads to meet wide-ranging customer needs.
- High scalability. Clouds are dynamically scalable to accommodate changing demands.
- On-demand. A cloud provides a huge resource pool from where on-demand resources can be provisioned. Cloud service usage can be metered similarly to how utilities like water, electricity, and gas are metered.
- Cost savings. With a broad selection of fault tolerance mechanisms available for clouds, service providers and enterprises can use inexpensive nodes to build their clouds. With automated, centralized cloud management, enterprises no longer need to grapple with the high costs entailed in managing a data center. By provisioning hardware-independent, general-purpose resources, cloud significantly improves resource utilization. These allow users to have quick access to cost-efficient cloud services and resources.

1.4.3 Development of Cloud Computing

There are three phases of cloud computing in terms of transforming the enterprise IT architecture from a legacy non-cloud architecture to a cloud-based one.

Cloud Computing 1.0

This phase deals with virtualization of IT infrastructure resources, with focus on compute virtualization. Enterprise IT applications are completely decoupled from the infrastructure. With virtualization and cluster scheduling software, multiple enterprise IT application instances and runtime environments (guest operating systems) can share the same infrastructure, leading to high resource utilization and efficiency. HCIA - Cloud Computing mainly covers the implementation and advantages of cloud computing in this phase.

Cloud Computing 2.0

Infrastructure resources are provisioned to cloud tenants and users as standardized services, and management is automated. These are made possible with the introduction of standard

service provisioning and resource scheduling automation software on the management plane, and software-defined storage and networking on the data plane. The request, release, and configuration of infrastructure resources, which previously required the intervention of data center administrators, are now fully automated, as long as the right prerequisites (e.g. sufficient resource quotas, no approval process in place) are met. This transformation greatly improves the speed and agility of infrastructure resource provisioning required by enterprise IT applications, and accelerates time to market (TTM) for applications by shortening the time needed to ready infrastructure resources. It transforms static, rolling planning of IT infrastructure resources into a dynamic, elastic, and on-demand resource allocation process. With it, enterprise IT is able to deliver higher agility for the enterprise's core applications, enabling it to quickly respond and adapt to changing demands. In this phase, the infrastructure resources provisioned to tenants can be virtual machines (VMs), containers (lightweight VMs), or physical machines. This transformation has not yet touched the enterprise's applications, middleware, or database software architectures that are above the infrastructure layer.

Cloud Computing 3.0

This phase is characterized by:

- A distributed, microservices-based enterprise application architecture.
- An enterprise data architecture redesigned using Internet technology and intelligence unleashed by big data.

In this phase, the enterprise application architecture gradually transforms from a vertical, hierarchical architecture that:

- Relies on traditional commercial databases and middleware suites
- Is purpose-designed for each application domain, siloed, highly sophisticated, stateful, and large scale

To

- A distributed, stateless architecture featuring lightweight, fully decoupled functionalities, and total separation of data and application logic
- Databases and middleware service platforms that are based on open-source yet enhanced enterprise-class architectures and fully shared across different application domains.

In this way, enterprise IT can deliver a new level of agility and intelligence for the enterprise business, further improve resource utilization, and lay a solid foundation for fast innovation in an iterative manner.

The majority of enterprises and industries have already passed Cloud Computing 1.0. Enterprises from some industries have already commercially deployed Cloud Computing 2.0, some on small scales though, and are now considering scaling up or continuing to move towards Cloud Computing 3.0. Others are moving from Cloud Computing 1.0 to 2.0, and are even considering implementing Cloud Computing 2.0 and 3.0 in parallel.

Copyright © Huawei Technologies Co., Ltd.

1.4.4 Further Reading: Differences Between Cloud Computing 1.0 and 2.0/3.0

This section talks about the major differences between Cloud Computing 1.0 and 2.0/3.0. The content comes from the book *Cloud Computing Architecture: Technologies and Practice*, written by Gu Jiongjiong.

Difference 1

From non-critical IT applications on cloud to telecom network applications and mission-critical enterprise applications on cloud. In the beginning, virtualization is used only for non-critical applications, such as desktop cloud and development & testing cloud. At this stage, applications are insensitive to the performance overheads caused by the virtualization software, and people's attention is mostly focused on the higher resource utilization and more efficient application deployment enabled by resource pooling. As cloud continues to grow in popularity, enterprises begin to move more business-critical applications, even their core production systems, to the cloud. Therefore, it has become crucial for a cloud platform to become more efficient and reliable in order to support critical enterprise applications that are mostly performance-demanding and latency-sensitive. Besides tangible assets such as compute, storage, and network resources, the most valuable asset of an enterprise is its data. In the compute virtualization phase of cloud computing, the front- and back-end I/O queues between the guest OS and host OS have high throughput overheads, while traditional structured data has demanding requirements on I/O throughput and latency. This is why in the beginning the part of the infrastructure that handles mission-critical structured data is excluded from the scope of virtualization and even resource pooling. However, as Xen and KVM-based virtualization engines keep improving I/O performance using techniques like single root input/output virtualization (SR-IOV) and multi-queue, virtualization platforms can now deliver the performance needed to run core enterprise applications, such as mission-critical relational databases like enterprise resource planning (ERP). In the recent two to three years, cloud computing has extended beyond the IT sector to penetrate the telecom sector. The success of the Internet, both in business model and technology, has encouraged telecom operators to reconstruct existing networks and network functions using cloud technology, so as to decouple telecom software from the proprietary hardware supplied by a limited choice of vendors, while also enjoying the benefits brought by the cloud, such as lower total cost of ownership (TCO) for hardware, energy savings, accelerated innovation and more efficient application deployment. A multinational operator running subsidiaries in different countries may be able to use cloud to enable fast, software-based customization of network functions and allow higher openness.

Difference 2

From compute virtualization to storage and network virtualization. The early phases of cloud computing focus on compute virtualization to support on-demand, elastic resource allocation and the decoupling of software from hardware. In fact, the now well-known compute virtualization technology can be traced back to the days of the IBM System/370, when it was first implemented on IBM mainframe computers. The idea was to put a virtualization layer between the OS and the bare metal hardware to simulate multiple runtime environments on top of the 370 mainframe instruction system. This led the upper layer applications to "believe" that they were running in a dedicated system. The compute virtualization engine enabled time-based CPU sharing between multiple virtual machines. In addition, access requests to memory, I/O, and network resources were also intercepted and proxied by the virtualization engine. Compute virtualization became widely available for commercial use only after x86-based servers became the mainstream IT hardware platforms, with different single-host compute virtualization implementations from VMware ESX, Xen, and KVM. Combined with

small- and medium-sized cluster management software (such as vCenter/vSphere, XenCenter, and FusionSphere) capable of dynamic VM migration and HA scheduling, these implementations have become the mainstream compute virtualization offerings. As data and information are becoming the core assets of enterprise IT, data storage media is split from servers to form a large, independent industry. Like the essential computing power provided by CPUs, storage plays an equally important role in data centers. Questions like "how to quickly accommodate enterprises' changing storage needs" and "how to make the best use of legacy storage systems supplied by multiple vendors" can best be answered by storage virtualization. Also, the hardware of modern data centers no longer merely consists of IBM mainframes or midrange computers deployed in master/slave mode. North-south traffic between the client and server, east-west traffic between different servers, and the communication between an enterprise's internal network and the public network all transmit through Ethernet and wide area networks (WAN) featuring peer-to-peer, open architectures. Therefore, network became the third essential element forming the IT infrastructure of a data center, along with compute and storage. In the context of an end-to-end data center infrastructure solution, server virtualization can no longer adequately support hardware-independent, elastic, and on-demand resource allocation. Instead, all three virtualization technologies, compute, storage, and network, must work together to get the job done. Besides the unified processing of API and information models by the cloud management and scheduling software on the management and control plane, one important characteristic of virtualization is to intercept the original access requests, extract keywords, and simulate the requested resources, though maybe in different granularities from the underlying physical resources. The CPU and memory resources from commodity x86 servers are virtualized into VM CPU and memory specifications and then provisioned to consumers (upper-layer users or applications) on an on-demand basis. With the rapid upgrade of compute components and the horizontal scalability enabled by a software-based load balancing mechanism, compute virtualization only need to deal with splitting resources into smaller units. However, for storage, due to limited single-disk capacities (SATA/SAS) in contrast with growing storage needs, it is necessary to aggregate the storage resources of multiple loosely-coupled and distributed servers across the entire data center, including both the disks inside servers and external SAN/NAS devices, to form a unified storage resource pool. This storage pool may be a homogeneous one consisting of storage software and hardware supplied by the same vendor, or it may consist of heterogeneous storage devices from multiple different vendors. All storage pools can be accessed through standard formats, such as block, object, and file storage. In a data center, network connectivity needs come from applications and are closely related to the compute and storage resources functioning as network nodes. However, traditionally, network switching functions were implemented on physical switches and routers. To upper-layer applications, network functions were just isolated "boxes" connected by communication links. These "boxes" could not sense the dynamic connectivity needs of upper-layer applications. Networking and isolation requirements from the service layer are accommodated entirely manually. In a multi-tenant virtualization environment, different tenants may have vastly different requirements on the configuration and management of edge routing and gateway devices. The built-in multi-instance feature of the physical routers and firewalls cannot meet the multi-tenancy requirements in the cloud environment either. On the other hand, deploying physical routers and firewalls to match existing tenants in quantity is simply not an economically viable option for most customers. This has led people to consider the possibility of migrating network functions from proprietary, close-architecture platforms to commodity x86 servers. In this way, instances can be created and destroyed on network nodes by the cloud OS platform in an automated manner. Virtual communication links, along with the necessary security isolation mechanisms, can be created between any two network nodes. The significance of this is that it enables application-driven, automated network management and configuration, significantly reducing the complexity of data center network management. From the perspective of resource utilization, the traffic between any two virtual network nodes needs to be exchanged over the underlying physical network. An unlimited number of virtual nodes can be created and used as long as they do not exceed the total

resource quotas of the physical network. (It is advisable to use the non-blocking Clos architecture for physical networks.) Network bandwidth resources are released as soon as the virtual network loads are released. This maximizes dynamic sharing of physical network resources. To sum up, network virtualization allows multiple "box-like" network entities to be presented as a unified network resource pool to upper layer applications, providing a unified collaboration mechanism for compute, storage, and network resources.

Difference 3

From internal services within an enterprise to multi-tenant infrastructure services and end-to-end IT services, with larger resource pools. An important goal of cloud computing is to allow users to use computing power as conveniently as using utilities like water and electricity. In the early days of cloud computing, virtualization technologies, such as VMware ESX, Microsoft Hyper-V, and Linux-based Xen and KVM, were widely used to implement server-centric virtualization and resource consolidation. At this stage, the servers in enterprise data centers, though virtualized and pooled, were still partially silos. The concepts of multi-tenancy and service automation were understood and accepted only by a few people. Server-centric resource pooling serves only the administration/management personnel for the data center infrastructure hardware and software. Before virtualization, data center management personnel manage servers, and storage and network devices. After virtualization, they manage VMs, storage volumes, software-based switches, and even software-based firewalls. This allows multiple application instances and OSs to share server resources to the maximum extent possible, increasing resource utilization by evening out peaks and troughs. In addition, extra high availability (HA) and fault tolerance (FT) mechanisms are provided for applications. Power usage effectiveness (PUE) is improved through dynamic resource scheduling and power management: aggregate light-load instances onto a small number of servers and split them when the load increases, and power off idle servers. However, virtualization only achieves higher resource utilization and PUE. It is still a long way from the real cloud featuring multi-tenancy and automated cloud service provisioning. So the natural next step following virtualization would be to extend the benefits of cloud computing to each tenant, rather than just the data center administrators. On top of the existing infrastructure O&M, monitoring, management portals, the cloud platform must be able to provision on-demand, customized infrastructure resources for each internal or external tenant by providing a portal for service subscription and daily maintenance and management or an API portal. Permissions like add, delete, modify, and search on virtual or physical resources must be delegated to each tenant while ensuring proper control and isolation. Each tenant is authorized to access only the compute and storage resources requested and created by themselves, along with the OS and application software bound with these resources. In this way, the tenants can have quick access to on-demand resources without purchasing any IT hardware equipment, and enjoy a new level of automation and agility brought by cloud. Thus, cloud benefits like the economy of scale and fast elasticity of on-demand resources can be fully exploited.

Difference 4

From small to huge datasets, and from structured data to unstructured or semi-structured data. With the increasing availability and popularity of smart devices, social networks, and the rise of the Internet of Things (IoT), the data transmitted over IT networks has changed from small-scale structured data to massive amounts of unstructured and semi-structured data, such as text, images, and videos. The data volumes have increased exponentially. The compute and storage capacities needed to process such massive amounts of unstructured and semi-structured data have far exceeded those that can be provided by traditional Scale-Up hardware architectures. Therefore, it has become imperative to fully utilize the Scale-Out architecture enabled by cloud computing, in order to create large-scale resource pools needed for mass data processing. The massive data sets accumulated during daily enterprise

Copyright © Huawei Technologies Co., Ltd.

transactions, as well as the data obtained from other sources, such as customer data from social networks or other websites, do not always require real-time processing. The data processing system does not need to be ready at all times. Therefore, a massive storage platform that is shared, plus the ability to dynamically allocate and release batch, parallel compute resources will be the most efficient means to support big data analytic needs.

Difference 5

From on-premises fixed-location computing and traditional man-machine interaction to cloud computing, smart mobile terminal, and immersive experience for enterprise and consumer applications. As enterprise and consumer applications are increasingly moved to the cloud, thin client as a way of accessing cloud services is gaining popularity. Compute and storage resources are also split from on-premises locations and moved to a remote data center for centralized deployment. In this circumstance, enterprises must find ways to guarantee consistent (sometimes immersive) user experiences for on-cloud applications regardless of where or how the access requests are initiated: using various thin clients or smart terminals from over a variety of networks, such as the enterprise's internal local area network (LAN), external fixed or mobile broadband, or wide area network (WAN). Facing problems like unstable performance and latency in packet forwarding, packet loss, and the absence of an end-to-end QoS mechanism for LAN or WAN communication, enterprises may find it exceptionally challenging to guarantee a cloud experience as good as or nearly as good as local computing. Cloud applications may be accessed using different methods and have different user experience standards. Common remote desktop protocols are becoming inadequate in delivering a user experience equivalent to that of local computing. Concerted efforts must be focused on optimizing the end-to-end QoS and QoE for IP multimedia (audio and video), and different methods may need to be used on the basis of dynamically identifying different types of workloads. Typical scenarios include the following:

- For common office applications, the response latency must be less than 100 ms, and the average bandwidth usage 150 kbps: GDI/ DX/OpenGL rendering instructions are intercepted on the server, and network traffic is monitored and analyzed in real time. Based on it, the optimal transmission methods and compression algorithms are selected, and the rendering instructions are redirected to thin clients or soft terminals, minimizing the latency and bandwidth usage.
- Virtual desktop usually delivers less-than-ideal VoIP quality. The default desktop protocol TCP is not suitable for VoIP. To solve this problem, RTP/UDP is used to replace TCP, and mature VoIP codec such as G.729/AMR is used. When VoIP/UC clients are used, VoIP traffic usually bypasses VMs to reduce the latency and voice quality overheads caused by additional coding and decoding. With these, the average mean opinion score (MOS) of voice services in virtual desktop scenarios has increased from 3.3 to 4.0.
- Remote access and playback of cloud-hosted HD (1080p/720p) videos: Common thin clients usually perform poorly in decoding HD videos. When there are concurrent access requests from multiple cloud desktops and media stream redirection is supported, the desktop protocol client software should have the ability to call the hard decoding capability of the thin client chip through a dedicated API. Some applications, such as Flash, and video software that directly reads from and writes to the GPU, rely on the concurrent codec capability of the GPU or hardware DSP. Software codec based on general-purpose CPUs will result in serious stalling problems and poor user experience. In this case, hardware GPU virtualization or DSP acceleration card can be used to improve the user experience of cloud-based HD video applications, delivering an HD and smooth viewer experience on par with that of local video playback. The desktop protocol should also be able to intelligently identify and differentiate the heat of image change and enable the bandwidth-intensive GPU data compression and redirection for

the image area that changes a lot, where the redirection of rendering instructions cannot keep up.

- Graph computing-intensive cloud applications, such as engineering or mechanical drawing, hardware PCB drawing, 3D games, and the most recent VR simulation: These applications also require large amounts of virtualized GPU resources for hardware-assisted rendering and compression acceleration. They also require significantly higher bandwidths: dozens to hundreds of Mbit/s bandwidth per channel, and 10 to 100 Gbit/s for concurrent access. The bandwidth between cloud access nodes and a centralized data center is finite. To provide higher and scalable bandwidth, one way is to split the large, centralized data center into multiple distributed data centers that are logically centralized but physically dispersed. This allows applications with a heavy load of human-machine interactions, such as VDI/VR, to be deployed at the Service PoPs in close proximity of end users.

The global consumer IT is entering to the post-PC era, and iOS and Android smart mobile devices are also gradually replacing PCs and even laptops used in offices. Enterprise users hope that with smart devices, they can access not only traditional Windows desktop applications, but also the enterprise's internal web SaaS applications, third-party SaaS applications, as well as other Linux desktop applications. They also hope that each cloud application can deliver consistent user experience, regardless of the user devices or OSs, and without the need of any adaptation efforts. The Web Desktop solution can meet all these requirements. It introduces the HTML5 protocol, and supports an OS that can run on multiple types of desktops, unified authentication and application aggregation, zero installation, upgrade, and maintenance for applications, delivering a consistent user experience across various types of smart devices.

Difference 6:

From homogeneous virtualization to heterogeneous virtualization, lightweight containers, and bare metal servers (physical machines). In the process of transforming from a traditional enterprise IT architecture to a modern, more agile one, enterprises need to be able to duplicate their applications quickly and in batches. Closed-source virtualization solutions offered by VMware and Hyper-V, and open-source ones such as Xen and KVM, are among the first to reach maturity for scale deployment. With virtualization, the best practices of application installation and configuration can be quickly duplicated in the form of VM templates and images, greatly simplifying repetitive yet sometimes complex installation, provisioning, and configuration of IT applications, reducing software deployment period to hours or even minutes. However, as enterprise IT applications increasingly transform from small-scale, monolithic, and stateful ones, to large-scale, distributed, and stateless ones with complete separation of data and application logic, people begin to realize that while VMs allow efficient resource sharing for multi-instance applications within large-scale IT data centers, they still cannot keep up with the elastic resource needs of applications like online web services and big data analytics. A single tenant may run multiple applications, with hundreds to thousands, or even tens of thousands of concurrent application instances. With VMs, an OS instance will need to be created for each application instance, leading to huge overheads. Also, the speed at which VM-based application instances can be created, started, and upgraded is still not fast enough for applications with unpredictable, massive resource needs. Internet companies like Google, Facebook, and Amazon have widely adopted Linux container technologies (such as namespace and cgroup). Based on the shared Linux kernel, these technologies isolate the runtime environments of application instances by containers. They also package and encapsulate the configuration information and runtime environment of each instance together, and use container cluster technology (such as Kubernetes, MESOS, and Swarm) to allow fast provisioning of highly concurrent, distributed multi-container instances and large-scale, dynamic orchestration and management of containers. This yields an unprecedented level of efficiency and agility in large-scale software deployment and life cycle

management, as well as iterative software development and rollout based on DevOps. In the long run, the container technology, being more lightweight and agile, will eventually replace virtualization. However, in a foreseeable future, container will continue to rely on cross-VM and -PM isolation mechanisms to isolate the runtime environments of different tenants and fulfills service level agreements. This is because container itself cannot properly address cross-tenant security isolation and excessive resource competition in the case of over-commitment of shared host resources, at least in the short term. Also, virtualization may not be suitable for some enterprise applications or middleware due to reasons like special vendor support policies, enterprise-class performance standards, and compatibility requirements. For example, virtualization cannot accommodate the performance needs of commercial databases, such as the Oracle RAC cluster database and HANA in-memory computing database. The customer wishes that these workloads, while running on physical machines, can also enjoy benefits like on-demand infrastructure resource provisioning based on resource pooling and automated configuration that are available for virtualization and containers. To achieve these, the cloud platform and cloud management software must not only automate OS and application installation on physical machines, but also enable collaborative management of storage and network resource pools and automated configuration while guaranteeing secure tenant isolation.

Difference 7

From closed-source and closed-architecture cloud platform and cloud management software to open-source and open-architecture ones. In the early stages of cloud computing, closed-source cloud platform software, such as VMware vSphere/vCenter and Microsoft's SystemCenter/Hyper-V was far more mature than open-source ones, and was therefore the firsthand choice for enterprises seeking to build their private clouds. With the rapid advancements of open-source communities and technologies, Xen and KVM-based cloud OSs, such as OpenStack, CloudStack, and Eucalyptus, are catching up, both in influence and market presence. Take OpenStack as an example. Many leading software and hardware companies, such as IBM, HP, SUSE, Red Hat, and Ubuntu, have become OpenStack platinum members. The community operates around a six-month, time-based release cycle with frequent development milestones. Since the release of the first version in 2010, a new OpenStack version has been released every six months, with active contributions from all community members. The functionalities of OpenStack community versions are fast and steadily iterating. In the first half of 2014, OpenStack had caught up with vCloud/vSphere 5.0 in terms of maturity and was ready for basic commercial deployment. Judging by the current situation and trends, OpenStack is becoming "the Linux in the cloud computing world." Around 2001, Linux OS was still relatively weak while Unix systems dominated most production platforms of enterprise IT systems. Back then, it would have been unimaginable that the open-source Linux would replace the closed-source Unix in just 10 years to have become the default choice of OS for enterprise IT servers. In addition, midrange computers and even mainframes are being replaced by commodity x86 servers.

1.5 Cloud Computing Models

Although not universally agreed upon, cloud computing is commonly categorized based on cloud deployment and service models.

1.5.1 By Deployment Model

Public Cloud

Public cloud is the earliest and best-known form of cloud computing. Our previous examples of cloud computing, including Backup & Restore on Huawei phones and Google Translate, are both examples of public cloud. Public cloud offers utility-like IT services over the Internet for the public.

Public clouds are usually built and run by cloud service providers. End users access cloud resources or services on a subscription basis while the service provider takes all the O&M and administration responsibilities.

Private Cloud

Private clouds are usually deployed for internal use within enterprises or other types of organizations. All the data of a private cloud is stored in the enterprise or organization's own data center. The data center's ingress firewalls control access to the data. A private cloud can be built based on the enterprise's legacy architecture, allowing most of the customer's hardware equipment to be reused. A private cloud may be able to deliver a higher level of data security and allow reuse of legacy equipment. However, these equipment will eventually need to be updated to keep up with growing demands, and doing so may entail high costs. On the other hand, stricter data access control also means less data sharing, even within the organization.

In recent years, a different type of private cloud has emerged encouraging enterprises to deploy core applications on the public cloud: Dedicated Cloud (DeC) on a public cloud. This model offers dedicated compute and storage resources and reliable network isolation, meeting the high reliability, performance, and security standards of tenants' mission-critical applications.

Hybrid Cloud

Hybrid cloud is a flexible cloud deployment mode. It may comprise two or more different types of clouds (public, private, and community, which we will talk about later) that remain distinctive entities. Cloud users can switch their workloads between different types of clouds as needed. Enterprises may choose to keep core data assets on-premises for maximum security while other data on public clouds for cost efficiency, hence a hybrid cloud model. With the pay-per-use model, public clouds offer a highly cost-efficient option for companies with seasonal data processing needs. For example, for some online retailers, their demands for computing power peak during holidays. Hybrid cloud also accommodates elasticity demands of other purposes, for example, disaster recovery. This means that a private cloud can use a public cloud as a disaster recovery destination and recover data from it when necessary. Another feasible option is to run applications on one public cloud while selecting another public cloud for disaster recovery purposes.

To sum up, a hybrid cloud allows users to enjoy the benefits of both public and private clouds. It also offers great portability for applications in a multi-cloud environment. In addition, the hybrid cloud model is cost-effective because enterprises can have on-demand access to cloud resources on a pay-per-use basis.

The downside is that hybrid cloud usually requires more complex setup and O&M. A major challenge facing hybrid cloud is integration between different cloud platforms, different types of data, and applications. A hybrid cloud may also need to address compatibility issues between heterogeneous infrastructure platforms.

Copyright © Huawei Technologies Co., Ltd.

Community Cloud

A community cloud is a cloud platform where the cloud infrastructure is built and managed by a leading organization of a specific community and shared between several organizations of that community. These organizations typically have common concerns, such as similar security, privacy, performance, and compliance requirements. The level of resource sharing may vary, and the services may be available with or without a fee.

Community cloud is not a new concept. Its difference from the public and private clouds lies in its industry attribute. For example, with a cloud built for the healthcare industry, patients' case files and records can be stored in this cloud. Doctors from every hospital can obtain patient information from the cloud for diagnostic purposes.

Community cloud can be a huge opportunity as well as a huge challenge. For example, with the community cloud for the healthcare industry, special efforts, including technical and administrative means, must be taken to ensure personal information security on the cloud.

1.5.2 By Service Model

In cloud computing, all deployed applications use some kind of hierarchical architecture. Typically, there is the user-facing interface, where the end users create and manage their own data; the underlying hardware resources; the OS on top of the hardware resources; and the middleware and application runtime environment on top of the OS. We call everything related to applications the software layer; the underlying, virtualized hardware resources (network, compute, and storage) the infrastructure; and the part in between the platform layer.

IaaS refers to a situation where a cloud service provider provides and manages the infrastructure layer while the consumer the other two layers. PaaS refers to a situation where the cloud service provider manages the infrastructure and platform layers while the consumer the application layer. SaaS means all three layers are managed by the provider.

Let's explain these three cloud service models using the example of a game.

Here are the *SEKIRO: SHADOWS DIE TWICE* System Requirements (Minimum)

- CPU: Intel Core i3-2100 | AMD FX-6300
- CPU SPEED: Info
- RAM: 4 GB
- OS: Windows 7 64-bit | Windows 8 64-bit | Windows 10 64-bit
- VIDEO CARD: NVIDIA GeForce GTX 760 | AMD Radeon HD 7950
- PIXEL SHADER: 5.0
- VERTEX SHADER: 5.0
- SOUND CARD: DirectX 11 Compatible
- FREE DISK SPACE: 25 GB
- DEDICATED VIDEO RAM: 2048 MB

Figure 1-10 Computer specifications requirements for a video game

The figure above shows the required computer specifications for this game. If we buy a computer of the required specifications, install an OS and then this game, this is not cloud computing. If we buy a cloud server of the same specifications from a public cloud provider, use an image to install an OS, download and then install the game, we're using the IaaS model. When installing a large game such as this one, we are likely to encounter the following error:

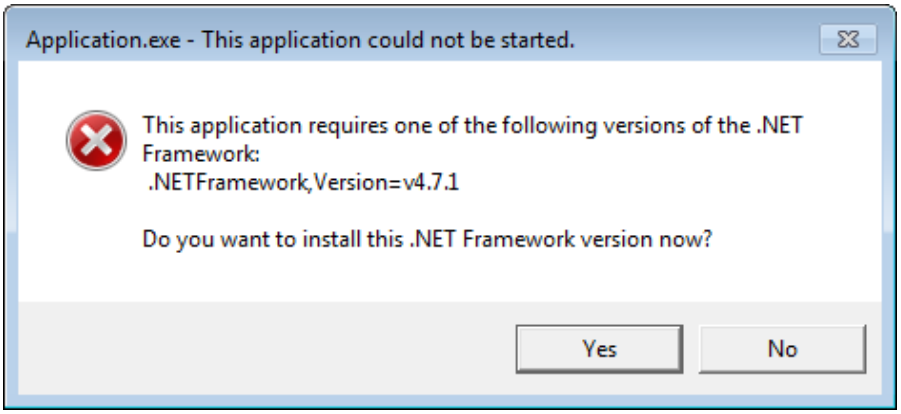


Figure 1-11 .NET Framework initialization error

This is because .NET Framework is part of the game's runtime environment. If we buy not only the cloud server but also the ready-to-go runtime environment with the .NET Framework already installed, we're using the PaaS model.

If we buy the cloud server with the game and all the necessary software already installed and all we need to do to start playing the game is to enter our user name and password, we're using the SaaS model.

2 Introduction to Compute Virtualization

Cloud computing 1.0 focuses on virtualization which today has become the foundation of cloud computing. Virtualization separates applications from the underlying operating systems (OSs) and hardware in traditional IT systems, and cloud computing relies on that split. While virtualization and cloud computing can both be used to build highly available and reliable runtime environments for applications, they differ in many ways.

The most apparent difference between the two is that virtualization provides only the Infrastructure as a Service (IaaS) model, while cloud computing provides other service models on top of IaaS. Virtualization is the foundation of cloud computing. It is important that we learn about this technology before delving into cloud technology.

Virtualization allows OSs and applications to run on virtual machines (VMs). This chapter describes how to create VMs and how virtualization works.

2.1 Virtualization Overview

2.1.1 What's Virtualization?

Virtualization is a technology that simulates hardware functionalities and creates multiple VMs on a physical server. Generally, applications need to run inside an OS, and only one OS can run on a physical server at a time. Virtualization allows VMs that reside on the same physical server to run independent OSs. This way, multiple OSs can concurrently run on the same physical server.

The essence of virtualization is to separate software from hardware by converting "physical" devices into "logical" folders or files.

Before virtualization, we can locate the physical devices running our applications in a real-world environment using an equipment list or physical configuration list. For example, in Figure 2-1, we can see several components of a physical server, like CPUs, memory, hard disks, and network adapters.

Figure 2-1 Physical server components

Virtualization converts physical servers into logical folders or files. These folders or files can be divided into two parts: those that store VM configuration information, and those that store user data. The following shows part of the configuration file of a KVM VM. We can see the VM name and its CPU, memory, hard disk, and network settings.

```
<name>instance-00000001</name>
  <uuid>0cc37c1c-3de2-416b-b291-7e3d612b1b39</uuid>
....................................
<memory unit='KiB'>8290304</memory>
<CPU mode='host-passthrough' check='none'>
    <topology sockets='4' cores='1' threads='1'/>
  </CPU>
....................................
<disk type='file' device='cdrom'>
     <driver name='qemu' type='raw' cache='none'/>
     <source
file='/opt/HUAWEI/image/instances/0cc37c1c-3de2-416b-b291-7e3d612b1b39/disk'/>
     <target dev='hda' bus='ide'/>
     <readonly/>
     <boot order='2'/>
     <address type='drive' controller='0' bus='0' target='0' unit='0'/>
</disk>
....................................
<interface type='bridge'>
     <mac address='fa:16:3e:b2:89:fa'/>
     <source bridge='plybac46573-e7'/>
     <virtualport type='openvswitch'>
       <parameters interfaceid='bac46573-e780-471b-b0c8-1193cabbf2bf'/>
     </virtualport>
     <target dev='tapbac46573-e7'/>
     <model type='virtio'/>
     <driver name='vhost'  vringbuf='4096'/>
     <boot order='3'/>
```

```
    <address type='pci' domain='0x0000' bus='0x04' slot='0x03' function='0x0'/>
  </interface>
```

Without virtualization, running multiple primary application programs in the same operating system of a physical server may cause runtime conflicts and performance bottlenecks. Running only one application on a dedicated server could solve these problems but will easily cause low resource utilization.

With virtualization, multiple VMs can run on a single physical server, and each VM can run an independent OS. This improves resource utilization. In addition, virtualization frees applications from being shackled to a single server by allowing dynamic VM migration across a cluster without impacting service continuity or user experience. Dynamic VM migration enables advanced features like high availability (HA), dynamic resource scheduling (DRS), and distributed power management (DPM), and brings a range of benefits for enterprise data centers, such as portability of workloads, server consolidation, fault tolerance, and lower OPEX and management costs. Chapter 5 introduces these features in detail.

2.1.2 A Brief History of Compute Virtualization

Virtualizing a physical server into multiple VMs is not a new technology in the IT industry. In fact, as early as 1964, "Big Blue" IBM started to virtualize mainframe computers. In 1961, IBM709 Machine realized a time-sharing system. The CPU is divided into a number of very short (1/100 sec) time slices, each performing a different task. By polling these time slices, you can virtualize or disguise a single CPU as multiple virtual CPUs, and make each virtual CPU appear to be running concurrently. This is the prototype of the VM. Later system360 machines all support time-sharing systems.

In 1972, IBM formally named the CTSS of the system370 machine a VM.

In 1990, IBM introduced the system390 machine, which supports logical partitioning. A CPU is divided into several parts (up to ten copies), and each CPU is independent. That is, a physical CPU can be logically divided into ten CPUs.

Until IBM put the time-sharing system open source, the personal PC finally came to the beginning of x86 virtualization. In 1999, VMware introduced the first x86 virtualization product.

While VMware develops its own virtualization products, Lan Pratt and Keir Fraser from University of Cambridge in London developed Xen VMs in a research project called Xenoserver in the 1990s. As the core of the Xenoserver, Xen VMs manage and allocate system resources and provide necessary statistics functions. In those days, x86 processors did not have hardware support for the virtualization technology and therefore Xen was introduced based on the quasi-virtualization technology. To support the running of multiple VMs, Xen required a modified kernel. Xen was officially open sourced in 2002 to allow a global community of developers to contribute and improve the product. Xen 1.0 was officially released followed a short time later by Xen 2.0. Widespread adoption of the Xen hypervisor took place when Red Hat, Novell, and Sun all added the Xen hypervisor as their virtualization solution of choice. In 2004, Intel's engineers began to develop hardware virtualization with Xen to provide software support for their next-generation processors. In 2005, Xen 3.0 was officially released, which supports Intel's VT technology and IA64 architecture, allowing Xen VMs to run unmodified operating systems.

In addition to Xen, Kernel-based Virtual Machine (KVM) is another famous virtualization technology, which was originally developed by Israeli startup Qumranet. KVMs were used as the VMs of their Virtual Desktop Infrastructure (VDI) products. To simplify development, KVM developers did not choose to write a new hypervisor. Instead, they loaded a new module based on the Linux kernel to make the Linux kernel become a hypervisor. For details about

hypervisors, see 2.1.3 "Compute Virtualization Types." In October 2006, after completing basic functions, dynamic migration, and optimization of main functions and performance, Qumranet officially announced the birth of KVM. Also in October 2006, the source code of the KVM module was officially accepted into the Linux kernel and became a part of the kernel source code. On September 4, 2008, Qumranet was acquired by Red Hat, Inc. for $107 million in cash. Red Hat is a famous Linux distribution vendor and a leading contributor to the kernel community. The acquisition means that Red Hat became the new owner of the KVM open source project. After the acquisition, Red Hat developed its own VM solution and began to replace Xen with KVM in its own products. In November 2010, Red Hat launched Red Hat Enterprise Linux 6 (RHEL 6), which integrated the latest KVM VM and deleted the Xen VM integrated in the RHEL 5.x series.

From 2006 to 2010, traditional IT vendors launched their own virtualization products. In 2007, HP launched Integrity VMs, and Microsoft added Hyper-V to Windows Server 2008 R2.

As x86 virtualization became more and more popular, a lightweight virtualization technology was also being developed, that is, container technology. The concept of containers was started way back in 1979 with UNIX chroot. In 2008, after years of development, Linux launched LXC. LXC stands for LinuX Containers and it is the first, most complete implementation of Linux container manager. It was implemented using cgroups and Linux namespaces. LXC was delivered in liblxc library and provided language bindings for the API in Python3, Python2, Lua, Go, Ruby, and Haskell. Contrast to other container technologies LXC works on vanila Linux kernel without requiring any patches. Today, the LXC project is sponsored by Canonical Ltd. and hosted here. In 2013, the Docker container project was developed. Docker also used LXC at the initial stages and later replaced LXC with its own library called libcontainer. Unlike any other container platform, Docker introduced an entire ecosystem for managing containers. This includes a highly efficient, layered container image model, a global and local container registries, a clean REST API, a CLI, etc. At a later stage, Docker also took an initiative to implement a container cluster management solution called Docker Swarm. In 2014, Rocket was launched, which is a much similar initiative to Docker started by CoreOS for fixing some of the drawbacks they found in Docker. CoreOS has mentioned that their aim is to provide more rigorous security and production requirements than Docker. More importantly, it is implemented on App Container specifications to be a more open standard.

2.1.3 Compute Virtualization Types

Before introducing compute virtualization types, let's learn some terms that are commonly used in virtualization.

First, a host machine is a physical computer that can run multiple VMs, and an OS installed and running on the host machine is a host OS. VMs running on a host machine are called guest machines. The OS installed on VMs is called a guest OS. The core of virtualization technology between the host OS and guest OS is a hypervisor, which is sometimes called Virtual Machine Manager (VMM).

In a physical architecture, a host has only two layers from bottom to top: hardware (host machine) and host OS. Applications are installed in the host OS. In a virtualization architecture, a host has more layers from bottom to top: hardware (host machine), hypervisor, guest machine, and guest OS. Applications are installed in the guest OS. Multiple guest machines can be created and run on a single host machine.

There are two types of hypervisors: Type 1 and Type 2. Many people categorize containers as the third type of hypervisor. Since containers are not discussed in this course, we will just focus on Type 1 and 2 hypervisors.

A Type 1 hypervisor is also called a bare-metal hypervisor. This type of hypervisor has direct access to hardware resources and does not need to access the host OS. The hypervisor can be

seen as a customized host OS, which merely functions as VMM and does not run other applications. The hypervisor provides the following basic functions: Identify, capture, and respond to privileged CPU instructions or protection instructions sent by VMs (the privileged instructions and protection instructions will be described in the CPU virtualization section); schedule VM queues and return physical hardware processing results to related VMs. In other words, the hypervisor manages all resources and virtual environments. VMM can be seen as a complete OS born for virtualization to control all resources (CPUs, memory, and I/O devices). VMM also provisions VMs for running the guest OS. Therefore, VMM also supports creation and management of virtual environments. The virtualization products that use Type 1 hypervisors include VMWare ESX Server, Citrix XenServer, and FusionCompute.

NOTE

In Type 1 virtualization, the hypervisor is dedicated to converting host resources into virtual resources for the guest OS to use. The guest OS runs as a process on the host. Therefore, such hypervisors are called bare-metal hypervisors.

Type 1 hypervisors have the following advantages and disadvantages:

- Advantages: VMs can run different types of guest OSs and applications independent of the host OS.
- Disadvantages: The kernel of the virtualization layer is hard to develop.

A Type 2 hypervisor is also called a hosted hypervisor. Physical resources are managed by the host OS (for example, Windows or Linux). VMM provides virtualization services and functions as a common application in the underlying OS (for example, Windows or Linux). VMs can be created using VMM to share underlying server resources. VMM obtains resources by calling the host OS services to virtualize the CPUs, memory, and I/O devices. After a VM is created, VMM usually schedules the VM as a process of the host OS. The virtualization products that use Type 2 hypervisors include VMware Workstation and Virtual PC.

Type 2 hypervisors have the following advantages and disadvantages:

- Advantages: They are easy to implement.
- Disadvantages: Only the applications supported by the host OS can be installed and used. The performance overheads are high.

NOTE

Unlike a Type 1 hypervisor, a Type 2 hypervisor is only a program in the host OS. All hardware resources are managed by the host OS.

Both Type 1 and Type 2 hypervisors possess the partitioning, isolation, encapsulation, and hardware independence features.

- Partitioning: indicates the VMM capability of allocating server resources to multiple VMs. Each VM can run an independent OS (same as or different from the OSs running on other VMs on the same server) so that multiple applications can coexist on one server. Each OS gains access only to its own virtual hardware (including the virtual NIC, virtual CPUs, and virtual memory) provided by VMM. The partitioning feature solves the following problems:
  - Resource quotas are allocated to each partition to prevent resource overuse by virtualization.
  - Each VM has an independent OS.
- Isolation: Multiple VMs created in a partition are logically isolated from each other. The isolation feature solves the following problems:
  - Even if one VM crashes due to an OS failure, application breakdown, or driver failure, it should not affect the others on the same server.

- It seems that each VM locates at an independent physical machine. If a VM is infected with worms or viruses, the worms and viruses are isolated from other VMs.

  The isolation feature allows you to control resources to implement performance isolation. That is, you can specify the minimum and maximum resource usages for each VM to prevent a VM from exclusively occupying all resources in the system. Multiple workloads, applications, or OSs can run on a single machine, without causing problems such as application conflicts and DLL conflicts mentioned in our discussions about the limitations of the traditional x86 architecture.

- Encapsulation: Each VM is saved as a group of hardware-independent files, including the hardware configuration, BIOS configuration, memory status, disk status, and CPU status. You can copy, save, and move a VM by copying only a few files. Let's use VMware Workstation as an example. You can copy a set of VM files to another computer where VMware Workstation is installed and restart the VM. Encapsulation is the most important feature for VM migration and virtualization. This is because encapsulating a VM as a set of hardware-independent files makes VM migration and hot swap possible.
- Hardware independence: After a VM is encapsulated into a group of files, the VM is completely independent from its underlying hardware. You can migrate the VM by copying the VM device file, configuration file, or disk file to another host. Because the underlying hardware device is shielded by VMM running on it, the migration can be successful as long as the same VMM running on the target host as that on the source host, regardless of the underlying hardware specifications and configuration. This is similar to editing a Word file by using Office 2007 on computer A that runs a Windows 7 system and then copying the Word file to computer B that runs a Windows 10 system. You only need to check whether Office 2007 is installed on computer B and do not need to check the CPU model or memory size of the underlying hardware.

2.2 Compute Virtualization

Compute virtualization includes CPU virtualization, memory virtualization, and I/O virtualization.

2.2.1 CPU Virtualization

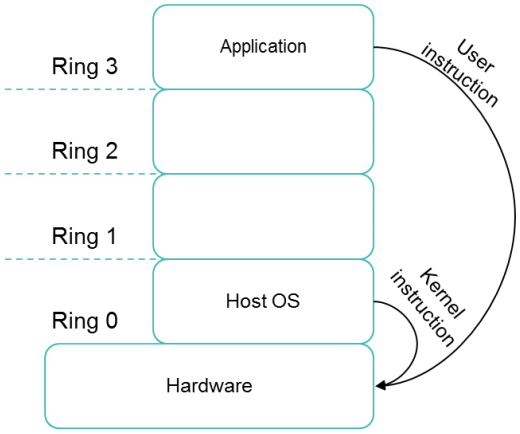


Figure 2-2 CPU hierarchical protection domains

Before we talk about CPU virtualization, let's have a brief look at the hierarchical protection domains of CPUs, often called protection rings. There are four rings: Ring 0, Ring 1, Ring 2, and Ring 3, which is a hierarchy of control from the most to least privilege. Ring 0 has direct

access to the hardware. Generally, only the OS and driver have this privilege. Ring3 has the least privilege. All programs have the privilege of Ring 3. To protect the computer, some dangerous instructions can only be executed by the OS, preventing malicious software from randomly calling hardware resources. For example, if a program needs to enable a camera, the program must request a Ring 0 driver to do that on its behalf. Otherwise, the operation will be rejected.

The OS on a common host sends two types of instructions: privileged instructions and common instructions.

- Privileged instructions are instructions used to manipulate and manage key system resources. These instructions can be executed by programs of the highest privilege level, that is, Ring 0.
- Common instructions can be executed by programs of the common privilege level, that is, Ring 3.

In a virtualized environment, there is another special type of instruction called sensitive instruction. A sensitive instruction is used for changing the operating mode of a VM or the state of a host machine. The instruction is handled by VMM after a privilege instruction that originally needs to be run in Ring 0 in the guest OS is deprived of the privilege.

The virtualization technology was firstly applied in IBM mainframes. How does mainframes implement CPU sharing? First, let's learn about the CPU virtualization methods of mainframes. The CPU virtualization methods used by mainframes are Deprivileging and Trap-and-Emulation, which are also called classical virtualization methods. The basic principles are as follows: The guest OS runs at the non-privilege level (that is, deprivileging) and VMM runs at the highest privilege level (that is, fully controlling system resources).

A problem arises: How can a privileged instruction sent by the guest OS of VM performed? Because the privilege of all VM systems has been removed, Trap-and-Emulation takes effect. After the privilege of the guest OS is removed, most instructions of the guest OS can still run on hardware. Only when a privileged instruction arrives, it will be sent to VMM for emulation. VMM, in place of the VM, sends the privileged instruction to the real hardware CPU.

Combining the classical CPU virtualization methods with the timer interrupt mechanism of the original OS can solve problems in CPU virtualization. For example, VM 1 sends privileged instruction 1 to VMM. An interrupt is triggered. VMM traps privileged instruction 1 sent by VM 1 for emulation, and then converts the instruction into privileged instruction 1' of the CPU. VMM schedules privileged instruction 1' to a hardware CPU for execution, and sends the result to VM 1, as shown in Figure 2-3. When VM 1 and VM 2 send a privileged instruction to VMM at the same time, the instruction is trapped into emulation, and VMM performs unified scheduling. Instruction 1' is executed and then instruction 2' is executed, as shown in Figure 2-4. The CPU virtualization function is successfully implemented by using the timer interrupt mechanism and the Deprivileging and Trap-and-Emulation methods.

Copyright © Huawei Technologies Co., Ltd.

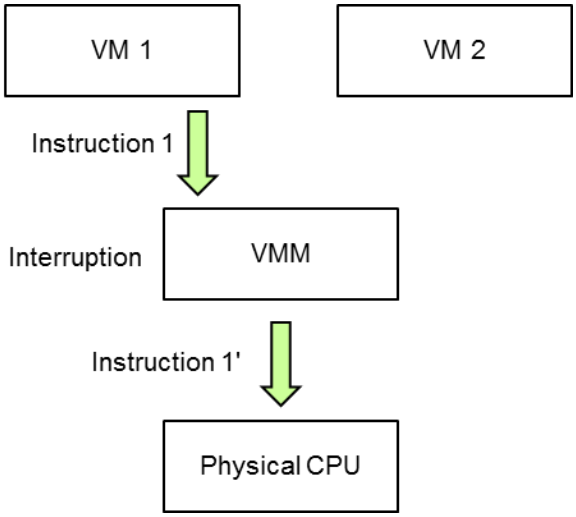


Figure 2-3 Unified scheduling of all instructions

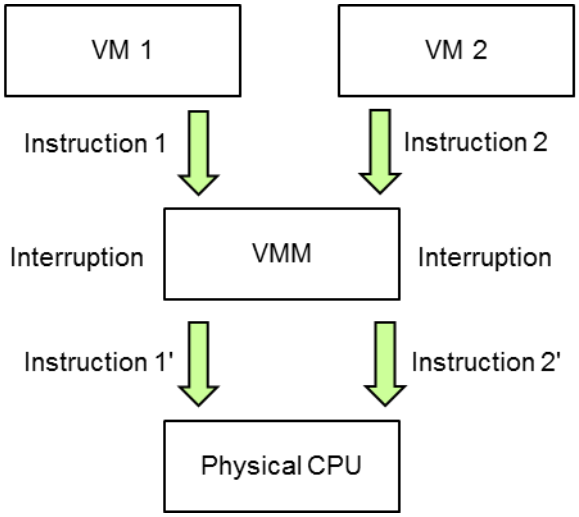


Figure 2-4 Special instructions

Why is the timer interruption mechanism required? If an emergency occurs outside the system, inside the system, or in the current program, the CPU immediately stops the running of the current program, and automatically switches to the corresponding processing program (interruption service program). After the processing is complete, the CPU returns to the original program. This process is called program interruption. For example, when you are watching a video, the instant messaging program suddenly displays a message, which triggers the interruption mechanism. The CPU will pause the video playing process and execute the instant messaging process. After processing the instant messaging operation, the CPU continues to execute the video playing process. Of course, the interruption time is very short and users are unaware of the interruption.

As the performance of x86 hosts is increasingly enhanced, applying virtualization technologies to the x86 architecture becomes a major problem for implementing x86 server virtualization. At this time, people usually think of the use of the CPU virtualization technology on mainframes. Can the CPU virtualization method used on mainframes be transplanted to x86 servers? The answer is no. But why? To answer this question, we need to understand the differences between the x86-architecture CPU and mainframe CPU.

Mainframes (including subsequent midrange computers) use the PowerPC architecture, that is, a reduced instruction set computer (RISC) architecture. In the CPU instruction set of the RISC architecture, sensitive instructions specific to VMs are included in privileged instructions, as shown in Figure 2-5. After the privilege of the VM OS is removed, the privileged instructions and sensitive instructions can be trapped, emulated, and executed. Because the privileged instructions include sensitive instructions, the CPU with the RISC architecture can properly use the Deprivileging and Trap-and-Emulation methods. However, CPU instruction sets in the x86 architecture are CISC instruction sets, which are different from RISC instruction sets, as shown in Figure 2-6.

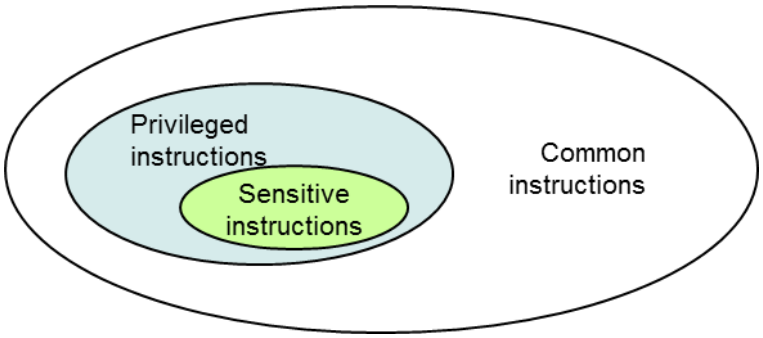


Figure 2-5 RISC instruction set

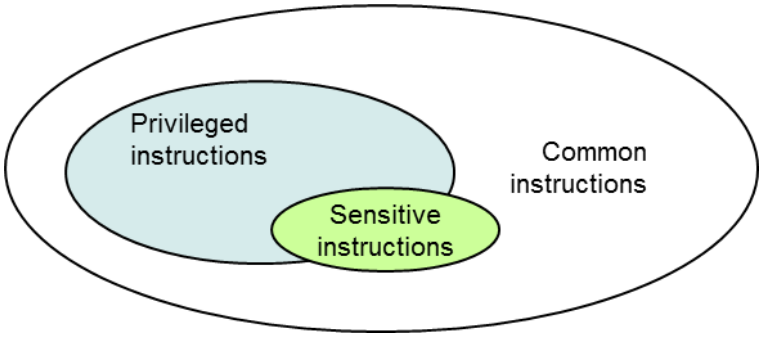


Figure 2-6 CISC instruction set

As shown in the preceding figures, the privileged instructions and sensitive instructions in CISC instruction set do not completely overlap. Specifically, 19 sensitive instructions in the CISC instruction set based on the x86 architecture are not privileged instructions. These sensitive instructions run in the Ring 1 user mode of the CPU. What problems will this bring about? Apparently, when a VM sends one of the 19 sensitive instructions, the instruction cannot be captured by VMM by means Trap-and-Emulation because it is not a privileged instruction. Therefore, x86 servers cannot be virtualized using the Deprivileging and Trap-and-Emulation methods. This problem is called a virtualization vulnerability. Since mainframe-based CPU virtualization methods cannot be directly transplanted to the x86 platform, what CPU virtualization methods should the x86 platform use?

IT architects came up with three alternative techniques. They are full virtualization, paravirtualization, and hardware-assisted virtualization (proposed by hardware vendors).

- Full virtualization

  The classical virtualization methods are not suitable for the x86-based CPUs. The root cause is that the 19 sensitive instructions beyond the privileged instructions. CPU virtualization problems can be solved only after these sensitive instructions can be

identified, trapped, and emulated by VMM. But how can these 19 instructions be identified?

A fuzzy identification method can be used. All OS requests sent by VMs are forwarded to VMM, and VMM performs binary translation on the requests. When VMM detects privileged or sensitive instructions, the requests are trapped into VMM for emulation. Then, the requests are scheduled to the CPU privilege level for execution. When VMM detects program instructions, the instructions are executed at the CPU non-privilege level. This technique is called full virtualization because all request instructions sent by VMs need to be filtered. Figure 2-7 shows the implementation of full virtualization.

Full virtualization was first proposed and implemented by VMware. VMM translates the binary code of the VM OS (guest OS) without modifying the VM OS. VMs have high portability and compatibility. However, binary translation causes the performance overhead of VMM. On one hand, full virtualization has the following advantages: The VM OS does not need to be modified. VMs are highly portable and compatible, and support a wide range of OSs. On the other hand, it has the following disadvantages: Modifying the guest OS binary code during running causes large performance loss and increases the VMM development complexity. Xen developed the paravirtualization technique, which compensates for the disadvantages of full virtualization.

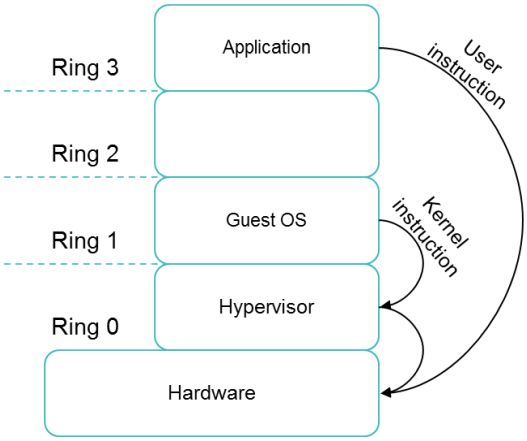


Figure 2-7 Full virtualization

- Paravirtualization

  The virtualization vulnerability comes from the 19 sensitive instructions. If we can modify the VM OS (guest OS) to avoid the virtualization vulnerability, then the problem can be solved.

  If the guest OS can be modified to be able to aware that it is virtualized, the VM OS uses the Hypercall to replace sensitive instructions in the virtualization with the hypervisor layer to implement virtualization. Non-sensitive instructions such as privileged and program instructions are directly executed at the CPU non-privilege level. Figure 2-8 shows paravirtualization. Paravirtualization has the following advantages: Multiple types of guest OSs can run at the same time. Paravirtualization delivers performance similar to that of the original non-virtualized system. Its disadvantages are as follows: The host OS can be modified only for open-source systems, such as Linux. Non-open-source systems, such as Windows, do not support paravirtualization. In addition, the modified guest OS has poor portability.

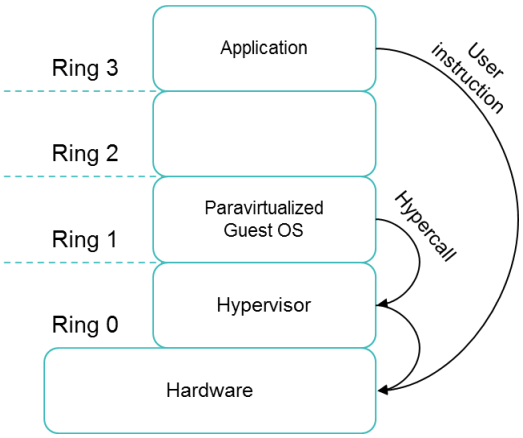


Figure 2-8 Paravirtualization

- Hardware-assisted virtualization

  In full virtualization and paravirtualization, the physical hardware does not support virtualization identification by default. If physical CPUs support virtualization and are able to identify sensitive instructions, it will be a revolutionary change to CPU virtualization.

  Fortunately, the CPUs of mainstream x86 hosts support hardware virtualization technologies, for example, Intel's Virtualization Technology (VT-x) CPU and AMD's AMD-V CPU. Intel's VT-x and AMD's AMD-V both target privileged instructions with a new CPU execution mode feature that allows VMM to run in a new ROOT mode below Ring 0. Privileged and sensitive calls are set to automatically trap to the hypervisor, removing the need for either full virtualization or paravirtualization. Hardware-assisted virtualization is used to solve virtualization vulnerabilities, simplify VMM software, and eliminate the need for paravirtualization or binary translation. Figure 2-9 shows the hardware-assisted virtualization technique.

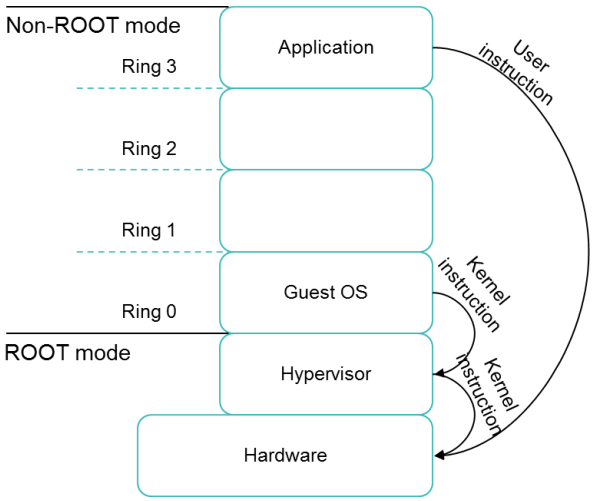


Figure 2-9 Hardware-assisted virtualization

2.2.2 Memory Virtualization

Memory virtualization is another important type of compute virtualization besides CPU virtualization. So why does CPU virtualization lead to memory virtualization?

With CPU virtualization, VMs running on top of the VMM layer have replaced physical hosts to run applications. Multiple VMs can run on the same host. A host usually has one or more memory modules. How can memory resources be allocated to multiple VMs properly? Memory virtualization was introduced to address this issue. One problem with memory virtualization is how to allocate memory address space. Generally, a physical host allocates memory address space as follows:

- The memory address starts from the physical address 0.
- The memory address space is allocated continuously.

However, after virtualization is introduced, the following problems occur: There is only one memory address space whose physical address is 0. Therefore, it is impossible to ensure that the memory address space of all VMs on the host start from the physical address 0. On the other hand, allocating contiguous physical addresses to VMs leads to low memory utilization and inflexibility.

Memory virtualization was introduced to solve problems concerning memory sharing and dynamical allocation of memory addresses. Memory virtualization is a process of centrally managing the physical memory of a physical machine and aggregating the physical memory into a virtualized memory pool available to VMs. Memory virtualization creates a new layer of address spaces, that is, the address spaces of VMs. The VMs are made to believe that they run in a real physical address space when in fact their access requests are relayed by VMM. VMM stores the mapping between guest machine address spaces and physical machine address spaces, as shown in Figure 2-10.

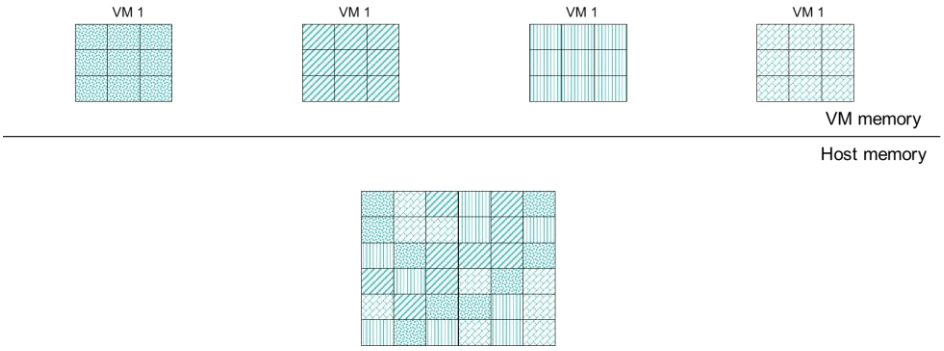


Figure 2-10 Memory virtualization

Memory virtualization involves the translation of three types of memory addresses: VM memory address (VA), physical memory address (PA), and machine memory address (MA). The following direct address translation path must be supported so that multiple VMs can run a physical host: VA (virtual memory) → PA (physical memory) → MA (machine memory). The VM OS controls the mapping from the virtual address to the physical address of the customer memory (VA → PA). However, the VM OS cannot directly access the machine memory. Therefore, the hypervisor needs to map the physical memory to the machine memory (PA → MA).

NOTE

We can use an example to explain the difference between MA and PA. If a server has a total of sixteen 16-GB memory bars, then its PA is 256 GB, and MA is sixteen memory bars distributed across different memory slots.

2.2.3 I/O Virtualization

With compute virtualization, a large number of VMs can be created on a single host, and these VMs all need to access the I/O devices of this host. However, I/O devices are limited. I/O device sharing among multiple VMs requires VMM. VMM intercepts access requests from VMs to I/O devices, simulates I/O devices using software, and responds to I/O requests. This way, multiple VMs can access I/O resources concurrently. I/O virtualization can be implemented in the following methods: full virtualization, paravirtualization, and hardware-assisted virtualization. Hardware-assisted virtualization is the mainstream technology for I/O virtualization.

- Full virtualization

  VMM virtualizes I/O devices for VMs. When a VM initiates an I/O request to an I/O device, VMM intercepts the request sent by the VM, and then sends the real access request to the physical device for processing. No matter which type of OS is used by the VM, the OS does not need to be modified for I/O virtualization. Multiple VMs can directly use the I/O device of the physical server. However, VMM needs to intercept I/O requests delivered by each VM in real time and emulates the request to a real I/O device. Real-time monitoring and emulation are implemented by software programs on the CPU, which causes severe performance loss to the server.

- Paravirtualization

  Unlike full virtualization, paravirtualization needs a privileged VM. Paravirtualization requires each VM to run a frontend driver. When VMs need to access an I/O device, the VMs send I/O requests to the privileged VM through the frontend driver, and the backend driver of the privileged VM collects the I/O request sent by each VM. Then, the backend driver processes multiple I/O requests by time and by channel. The privileged VM runs the physical I/O device driver and sends the I/O request to the physical I/O device. After processing the request, the I/O device returns the processing result to the privileged VM. VMs send I/O requests to a privileged VM and then the privileged VM accesses a real I/O device. This reduces the performance loss of VMM. However, the VM OS needs to be modified. Specifically, the I/O request processing method of the OS needs to be changed so that all the I/O requests can be sent to the privileged VM for processing. This requires that the VM OS can be modified (usually Linux). Figure 2-11 shows the Xen architecture.

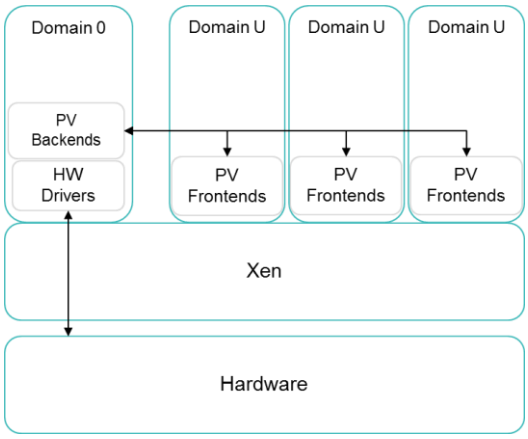


Figure 2-11 Xen architecture

In Figure 2-11, Domain 0 is a privileged VM, and Domain U is a user VM. The device information of all user VMs is stored in the XenSToRe of the privileged VM Domain0. The XenBus (a paravirtualization driver developed for Xen) in the user VM communicates with the XenSToRe of the Domain 0 to obtain the device information and

load the frontend driver corresponding to the device. When the user VM sends an I/O request, the frontend device driver forwards all data to the backend driver through the interface. The backend driver processes the data of the I/O request by time and by channel. Finally, the physical I/O device driver of Domain 0 sends the I/O request to the physical I/O device.

Let's take an example to compare full virtualization and paravirtualization. In full virtualization, VMM acts as an investigator. It collects and summarizes opinions and requests of each customer (VM). In paravirtualization, an opinion receiving box (that is, a privileged VM) is prepared and each customer puts opinions and requests into the box, and VMM centrally processes the opinions and requests. Paravirtualization significantly reduces the performance loss of VMM and therefore delivers better I/O performance. Full virtualization and paravirtualization have similarities: VMM is responsible for I/O access processing, which causes performance loss when VMs access I/O devices.

- Hardware-assisted virtualization

  Different from the preceding two methods, hardware-assisted virtualization directly installs the I/O device driver in the VM OS without any change to the OS. This method is equivalent to traditional PC OS access to hardware. Therefore, the time required for a VM to access the I/O hardware is the same as that for a traditional PC to access the I/O hardware. In the preceding example, hardware-assisted virtualization is like an intelligent information collection and processing platform. Users' requests can be directly submitted to the platform and the platform automatically processes the requests. Therefore, hardware-assisted virtualization outperforms full virtualization and paravirtualization in terms of I/O performance. However, hardware-assisted virtualization requires special hardware support.

2.2.4 Mainstream Compute Virtualization

CPU virtualization, memory virtualization, and I/O virtualization can be implemented to enable the reuse of physical resources. Multiple virtual servers can run on a physical host at the same time, and each virtual server can run different workloads. This improves hardware utilization. In addition, everything about a virtual server can be packed into a single file or folder. This breaks the tight coupling between software and hardware and allows VMs to migrate across hosts and even data centers, improving the reliability of workloads running on VMs. In cloud computing, we mainly use virtualization to implement IaaS cloud services.

There are three cloud service models: IaaS, PaaS, and SaaS. Some PaaS and SaaS services are implemented based on virtualization, and some are implemented based on physical hardware and distributed computing.

Let's use the first Chinese hard science fiction movie "The Wandering Earth" as an example, which shocked many people owing to its vivid pictures. The movie pictures were rendered using the rendering solution of the Huawei public cloud (HUAWEI CLOUD) that involves multiple products. SD rendering can be implemented by the C3 ECS and other cloud services. The C3 ECS uses the virtualization technology at the bottom layer. Panoramic rendering can be implemented by the BMS and other cloud services. The BMS background uses the real physical server instead of the virtualization technology.

Cloud computing is a business model that provides users with IT services anytime anywhere. Virtualization is an important technical means for cloud computing implementation.

There are many mainstream virtualization technologies. Generally, open-source and closed-source are used for classification. Open-source technologies include KVM and Xen. Closed-source virtualization technologies include Microsoft Hyper-V, VMware vSphere, and Huawei FusionSphere.

Copyright © Huawei Technologies Co., Ltd.

Open-source technologies are free of charge and can be used anytime. Users can customize some special requirements based on open-source code. Open-source technologies have high requirements on users' technologies. Once a problem occurs in the system, the system recovery strongly relies on the administrator's skillset and experience. In closed-source technologies, users cannot view or customize source code. Closed-source virtualization products are generally not free of charge and can be used out of the box. If a system problem occurs, vendors provide all-round support.

For users, it is meaningless to determine which is better between open-source or closed-source virtualization technologies, but determining their respective application scenarios makes senses.

In open-source virtualization technologies, Xen is on a par with KVM. KVM is full virtualization, while Xen supports both paravirtualization and full virtualization. KVM, a module in the Linux kernel, is used to virtualize CPUs and memory. It is a process of the Linux OS. Other I/O devices (such as NICs and disks) need to be virtualized by QEMU. Different from KVM, Xen directly runs on hardware, and VMs run on Xen. VMs in Xen are classified as the privileged VM (Domain 0) that has the permission to directly access hardware and manage other VMs (for example, Domain U). Domain 0 must be started before other VMs. Domain U is a common VM and cannot directly access hardware resources. All operations on Domain U must be forwarded to Domain 0 through frontend and backend drivers. Domain 0 completes the operations and returns the results to Domain U.

2.3 KVM

Huawei virtualization products earlier than the 6.3 version are developed based on Xen. In 6.3 and later, they are developed based on Kernel-based Virtual Machine (KVM).

KVM is a Type-II full virtualization solution. It is a Linux kernel module. A physical machine with a Linux kernel module installed can function as a hypervisor, which does not affect the other applications running on the Linux OS. Each VM is one or more processes. You can run the kill command to kill the processes.

After the KVM module is installed in a common Linux OS, three running modes are added:

- Guest Mode: VMs, including their CPUs, memory, and disks, run in a restricted CPU mode.
- User Mode: The quick emulator (QEMU) typically runs in this mode. QEMU emulates I/O requests.
- Kernel Mode: In this mode, the hardware can be operated. When the guest OS executes an I/O operation or privileged instruction, a request needs to be submitted to the user mode, and then the user mode initiates a hardware operation request to the kernel mode again to operate the hardware.

A KVM system consists of three parts: KVM kernel module, QEMU, and management tool. The KVM kernel module and QEMU are the core components of KVM, as shown in Figure 2-12.

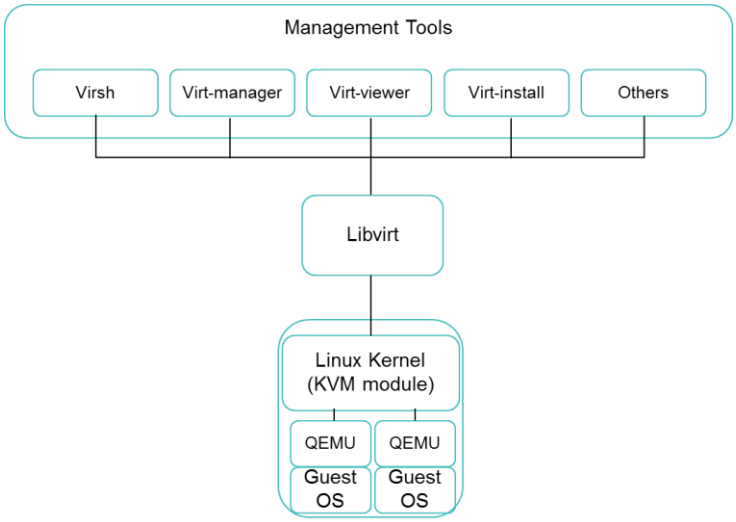


Figure 2-12 KVM architecture

Other virtualization products use similar architectures.

The KVM kernel module is the core of a KVM VM. This module initializes the CPU hardware, enables the virtualization mode, runs the guest machine in the VM mode, and supports the running of the virtual client.

KVM is composed of two kernel modules, a common module (kvm.ko) and a processor specific module (kvm-intel.ko or kvm-amd.ko). kvm.ko implements the virtualization functions. By itself, KVM does not perform any emulation. Instead, it exposes the /dev/kvm interface, which a userspace host can then use to create vCPUs, allocate address space for virtual memory, read and write vCPU registers, and run vCPUs. kvm.ko only provides CPU and memory virtualization. However, a VM requires other I/O devices such as NICs and hard disks besides CPUs and memory. QEMU is required to implement other virtualization functions. **Therefore, the KVM kernel module and QEMU form a complete virtualization technology.**

QEMU was not a part of KVM. It was a universal open-source virtualization emulator that uses pure software to implement virtualization. The guest OS considers that it is interacting with hardware. Actually, QEMU is interacting with hardware. This means that all interactions with the hardware need to pass through QEMU. Therefore, the simulation performance delivered by QEMU is low. QMEU is able to simulate CPUs and memory. In KVM, only QEMU is used to simulate I/O devices. KVM developers reconstructed QEMU to create QEMU-KVM.

In QEMU-KVM, KVM runs in the kernel space, and QEMU runs in the user space. When the guest OS issues instructions, the CPU- and memory-related instructions call the /dev/kvm interface through /ioctl in QEMU-KVM. In this way, the instructions are sent to the kernel module. From the perspective of QEMU, this is done to accelerate virtualization. Other I/O operations are implemented by the QEMU part in QEMU-KVM. KVM and QEMU form the complete virtualization technology.

In addition to virtualization of various devices, QEMU-KVM provides native tools for creating, modifying, and deleting VMs. However, Libvirt is the most widely used tool and API for managing KVM VMs.

Libvirt is an open-source project and is a powerful management tool. It is able to manage virtualization platforms such as KVM, Xen, VMware, and Hyper-V. Libvirt is an API developed using the C language. APIs developed using other languages, such as Java, Python,

and Perl, can call the Libvirt API to manage virtualization platforms. Libvirt is used by many applications. In addition to the virsh command set, Virt-manager, Virt-viewer, and Virt-install can manage KVM VMs through Libvirt.

In cloud computing, there are various hypervisors. Each hypervisor has its own management tool, and parameters are complex and difficult to use. Hypervisors are not unified, and there is no unified programming interface to manage them, which severely affects the cloud computing environment. With Libvirt, it can connect to various hypervisors, such as KVM and Xen, and provide APIs in various languages. Libvirt serves as the middle layer between the management tool and hypervisor and is completely transparent to upper-layer users.

QEMU is an emulation software tool for implementing I/O virtualization. It has poor emulation performance. For example, if QEMU is used to simulate a NIC of a Windows VM, the NIC rate displayed on the system is only 100 Mbit/s. It cannot meet the high NIC rate requirements of some applications. A new technology, Virtio, was introduced. In Windows virtualization, using Virtio can increase the NIC rate of a Windows VM to 10 Gbit/s.

Let's see how VM disk operations are performed without Virtio.

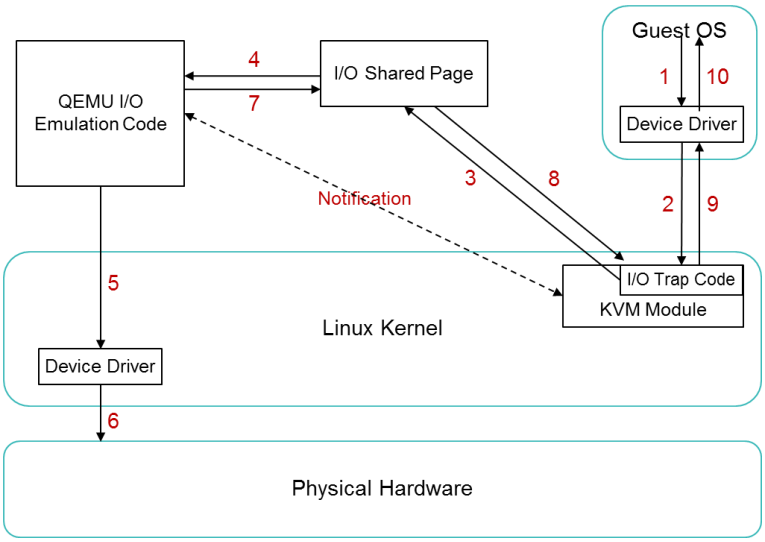


Figure 2-13 Default I/O operation process

1. A disk device of a VM initiates an I/O operation request.
2. I/O Trap Code (I/O capture program) in the KVM module captures the I/O operation request, performs corresponding processing, and then puts the processed request into the I/O shared page.
3. The KVM module notifies QEMU that a new I/O operation request is placed in the shared page.
4. After receiving the notification, QEMU obtains the detailed information about the I/O operation request from the shared page.
5. QEMU simulates the request and calls the device driver running in kernel mode based on the request information to perform the real I/O operation.
6. The I/O operation is then performed on physical hardware through the device driver.
7. QEMU returns the operation result to the shared page and notifies the KVM module that the I/O operation is complete.
8. I/O Trap Code reads the returned result from the shared page.
9. I/O Trap Code returns the operation result to the VM.

10. The VM returns the result to the application that initiated the operation.

NOTE

In steps 2, 3, and 7, KVM does not make any modification on the I/O operation except for capturing the request and sending the notification. The Virtio technology was developed to simply this procedure.

If Virtio is used, the procedure is as follows:

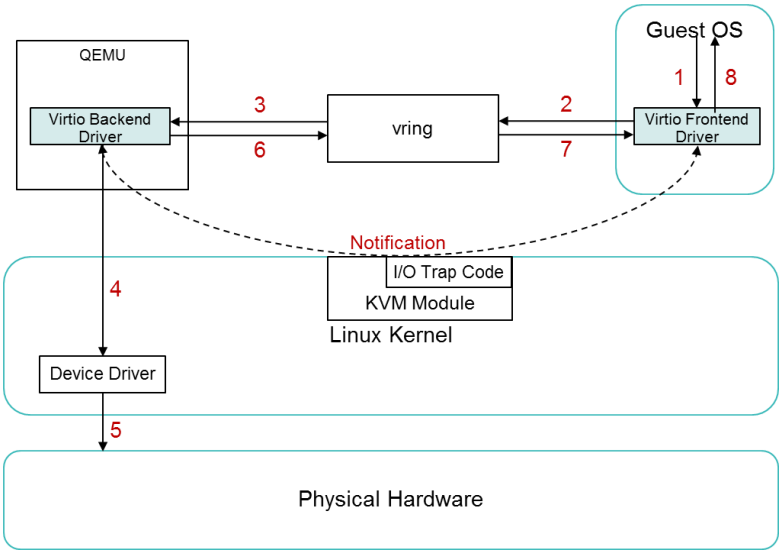


Figure 2-14 I/O operation process with Virtio used

1. The VM initiates an I/O operation request.
2. The I/O operation request is not captured by the I/O capture program. Instead, the request is stored in the ring buffer between the frontend and backend drivers. At the same time, the KVM module notifies the backend driver.
3. QEMU obtains the detailed information about the operation request from the ring buffer.
4. The backend driver directly calls the actual physical device driver to perform the I/O operation.
5. The operation is completed by the device driver.
6. QEMU returns the operation result to the ring buffer, and the KVM module notifies the frontend driver.
7. The frontend driver obtains the operation result from the ring buffer.
8. The frontend driver returns the result to the application that initiated the operation.

The advantages of Virtio are as follows:

- Saves the hardware resources required for QEMU emulation.
- Reduces the number of I/O request paths and improves the performance of virtualization devices.

Virtio has some disadvantages. For example, some old or uncommon devices cannot use Virtio but can only use QEMU.

2.4 FusionCompute

Huawei FusionSphere virtualization suite is a leading virtualization solution. This solution significantly improves data center infrastructure efficiency and provides the following benefits for customers:

- Improves infrastructure resource utilization data centers.
- Significantly accelerates service rollout.
- Substantially reduces power consumption in data centers.
- Provides rapid automatic fault recovery for services, decreases data center costs, and increases system runtime by leveraging high availability and powerful restoration capabilities of virtualized infrastructure.

The FusionSphere virtualization suite virtualizes hardware resources using the virtualization software deployed on physical servers, so that one physical server can function as multiple virtual servers. It consolidates existing VMs on light-load servers to maximize resource utilization and release more servers to carry new applications and solutions.

FusionCompute is the cloud OS software in the FusionSphere virtualization suite and a mandatory component. It virtualizes hardware resources and centrally manages virtual resources, service resources, and user resources. It virtualizes compute, storage, and network resources using the virtual computing, virtual storage, and virtual network technologies. It centrally schedules and manages virtual resources over unified interfaces. FusionCompute provides high system security and reliability and reduces the OPEX, helping carriers and enterprises build secure, green, and energy-saving data centers.

FusionCompute consists of two parts: Computing Node Agent (CNA) and Virtual Resource Manager (VRM). In addition to CNA and VRM, Unified Virtualization Platform (UVP) is a unified virtualization platform developed by Huawei. UVP is a hypervisor, like KVM and Xen. The FusionCompute hypervisor adopts the bare-metal architecture and can run directly on servers to virtualize hardware resources. With the bare-metal architecture, FusionCompute delivers VMs with almost server-level performance, reliability, and scalability.

The architecture of FusionCompute is similar to that of KVM. VRM functions as the management tool of KVM. Administrators and common users can manage and use FusionCompute on the GUI-based portal of VRM. VRM is based on the Linux OS. Therefore, many Linux commands can be used after you log in to VRM.

VRM provides the following functions:

- Manages block storage resources in a cluster.
- Manages network resources, such as IP addresses and virtual local area network (VLAN) IDs, in a cluster and allocates IP addresses to VMs.
- Manages the lifecycle of VMs in a cluster and distributes and migrates VMs across compute nodes.
- Dynamically scales resources in a cluster.
- Implements centralized management of virtual resources and user data and provides elastic computing, storage, and IP address services.
- Allows O&M personnel to remotely access FusionCompute through a unified web UI to perform O&M on the entire system, such as resource monitoring, resource management, and resource report query.

CNA is similar to the QEMU+KVM module in KVM. CNA provides the virtualization function. It is deployed in a cluster to virtualize compute, storage, and network resources in the cluster into a resource pool for users to use. CNA is also based on the Linux OS.

Copyright © Huawei Technologies Co., Ltd.

CNA provides the following functions:

- Provides the virtual computing function.
- Manages the VMs running on compute nodes.
- Manages compute, storage, and network resources on compute nodes.

CNA manages VMs and resources on the local node. VRM manages clusters or resources in the resource pool. When users modify a VM or perform other VM lifecycle operations on VRM, VRM sends a command to the CNA node. Then, the CNA node executes the command. After the operation is complete, CNA returns the result to VRM, and VRM records the result in its database. Therefore, do not modify VMs or other resources on CNA. Otherwise, the records in the VRM database may be inconsistent with the actual operations.

In addition to Huawei's hardware products, FusionCompute also supports other servers based on the x86 hardware platform and is compatible with multiple types of storage devices, allowing enterprises flexibly choose appropriate devices. A cluster supports a maximum of 64 hosts and 3000 VMs. FusionCompute provides comprehensive rights management functions, allowing authorized users to manage system resources based on their specific roles and assigned permissions.

This course uses FusionCompute to experiment virtualization functions and features. For details, see the corresponding experiment manual.

After the installation is completed based on the experiment manual, we will verify the following items:

1. Is FusionCompute 6.3 developed based on KVM?
2. If it is, does it use QEMU and Libvirt?

3 Network Basics for Cloud Computing

A network consists of various types of network devices. In a traditional IT system, nearly all network devices are physical devices with predictable traffic. For example, to enable the hosts connected by two different switches to communicate with one another, the two switches must be connected using a network cable or optical fiber. With cloud computing, many network devices are virtual entities running inside servers. These virtual devices communicate with each other not directly through a network table, but through entries in some logical forwarding table, which poses new challenges to network administrators. This chapter describes the physical and virtual devices used in cloud environments.

3.1 Network Architecture in Virtualization

3.1.1 Traffic on a Virtual Network

In a cloud or virtualization-based environment, traffic can be divided into north-south traffic and east-west traffic, while a traditional IT system does not make such distinction. A reference object is required to define north-south and east-west traffic. Generally, the router acts as the demarcation point distinguishing north-south traffic from east-west traffic. The router can be a physical or virtual one. Traffic that passes through a router is north-south traffic, while traffic that does not pass through a router is east-west traffic. The following figure shows north-south and east-west traffic in a system where physical routers are deployed on the edge of an Internet Data Center (IDC). In the north, the routers are connected to an extranet, which can be the Internet or an enterprise-defined extranet. In the south, the routers are connected to business networks, such as the email system and the office system of the IDC. When the extranet accesses a business network of the IDC, north-south traffic is produced. When an employee VM in the IDC accesses a business network, east-west traffic is produced because the traffic passes through no router.

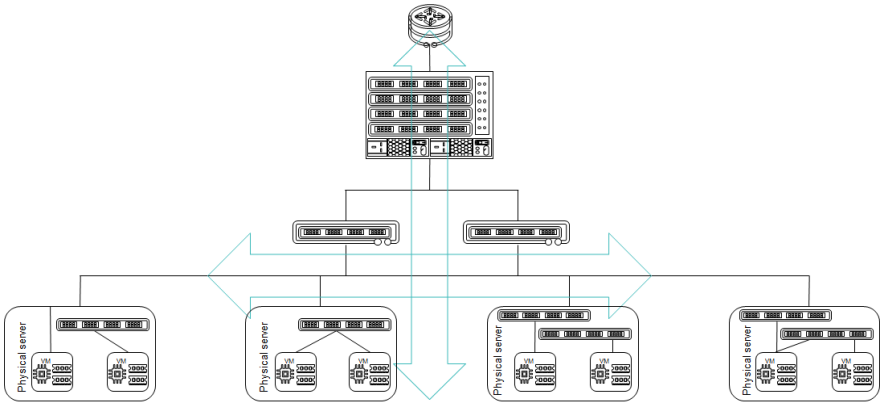


Figure 3-1 Traffic in a cloud environment

As cloud computing develops further, even more computations are performed in clusters of the IDC. A single request initiated by a user may trigger many computing operations between IDC business networks. If a cloud desktop is adopted, the client is also deployed in the IDC. In addition, virtualization features high availability (HA), which allows VMs to migrate across physical servers in the IDC. As a result, east-west traffic has increased from the original some 20% to the current 70%. This also creates a new network architecture. The old architecture consists of the core layer, the aggregation layer, and the access layer while the new architecture is a large layer-2 system. The old 3-layer architecture is not suitable for the data exchange in a cloud environment while the large layer-2 architecture handles massive east-west traffic with ease.

The VM is also a component of the network. VMs are almost all connected to the network using network bridges. To facilitate management and configuration, the virtual switch is used in most cases. The virtual switch is an advanced network bridge and will be described in detail later. A VM can have multiple virtual NICs. Through virtual NICs, a VM can connect to one or more virtual switches.

3.1.2 Basic Network Concepts

Broadcast and Unicast

Broadcast and unicast are two types of network traffic. There is a third traffic type called multicast, which is not discussed in this document.

Broadcast, as its name implies, means "one speaks and all others listen". On a network, when two devices intend to communicate with each other for the first time, the initiator will broadcast packets to identify receivers. These packets are flooded across the entire broadcast domain, and all network devices in the domain will receive the packets. When a network device receives the packets, it checks their content. If the network device finds itself a target receiver, it responds to the initiator with a unicast message. If the network device finds itself not a target receiver, it discards the packets.

Broadcast is not only used for starting communication but also used for running many applications including DHCP. A service uses a specific broadcast address. For example, the broadcast address **192.168.1.255** is used for the IP address range **192.168.1.0/24**, and the DHCP client uses the broadcast address **255.255.255.255** to search for DHCP servers on the initial stage.

An analogy may help you understand it better. Years ago, each village was usually equipped with a broadcast station. When the village intended to find somebody, an employee of the broadcast station would speak before the speaker, and the voice of the employee can be seen

as a broadcast address. When the village intended to publish important information, the village head would speak before the speaker, and the voice of the head can be seen as a broadcast address. Each type of event was broadcast by a different person, whose voice is unique.

Unicast differs from broadcast in that one person speaks while another person listens. On a real network, unicast communication is performed between two network devices. If one sends messages and the other receives messages, the process is half duplex. If both of them send and receive messages simultaneously, the process is full duplex.

On a network, data is usually transmitted in unicast mode. For example, to send an email, visit a web page, or play an online game, you need to establish a connection with the email server, web server, or game server.

Broadcast packets usually contain only basic information, such as the source and destination addresses. Unicast packets, however, usually contain information vital to both ends. If broadcast packets are flooded over a network and the bandwidth size is limited, unicast packets will be blocked. In this case, users may find that web pages cannot be displayed, emails cannot be sent, and their game accounts go offline frequently. Broadcast packets consume extensive bandwidths and pose security risks. Specifically, broadcast packets will be received by all, and if someone uses the information carried in the packets for fraud and spoofing, it may incur information leakage or network breakdown. Broadcast is necessary because it is the necessary step to start unicast. To mitigate the risks imposed by broadcast, network designers confine broadcast packets in a broadcast domain. Then, a question arises: Can we make a broadcast domain smaller? A smaller broadcast domain will relieve network congestion and mitigate security risks. Another question is through what devices broadcast domains communicate with each other. Next, we will consider the route and the default gateway.

Route and Default Gateway

Before mobile phones were widely used, telephones prevailed. When making a distance call using a telephone, you need to add an area code before the telephone number. To obtain an area code, you may need to consult yellow pages or other sources. An area code acts as a route, routing phone calls to the corresponding area, and the yellow pages act as a route table containing routes. If communication between two broadcast domains is like a distance call, the way a broadcast domain locates another broadcast domain is like a routing process.

When many broadcast domains exist, the route table will have a large number of entries. Each time a device attempts to communicate with another device, it needs to search for the route to the destination in the route table. This burdens the device where the route table is stored and reduces network communication efficiency. To address these issues, the default gateway is used. The default gateway works like the default route. An analogy may help you understand it better. When you need to obtain a target telephone number, you can dial **114**. The default gateway provides a function similar to what **114** provides, but there is a difference. After a user dials **114** to ask for a telephone number and finally obtains the number, the user dials the number themselves. After a default gateway receives a communication request, it performs route forwarding for the initiator if its route table contains the destination address, and it returns a message of an inaccessible destination address to the initiator if its route table does not contain the destination address.

The default gateway is a special routing mechanism. It is the last choice for route forwarding, which means that the default gateway is used for forwarding when no routing entry is available for forwarding.

VLAN

Virtual local area network (VLAN) is the most common way to subdivide a physical local area network (LAN) into separate logical broadcast domains. Devices within the same VLAN can communicate with each other using their MAC addresses, which those in different VLANs cannot. Hence, broadcast packets can be propagated only within a VLAN, not across VLANs. For details, see the following figure.

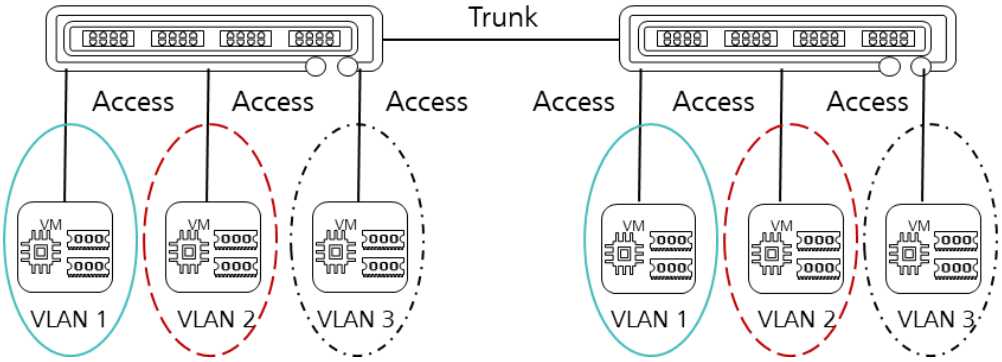


Figure 3-2 VLAN functions

VLAN brings the following benefits:

- Defines broadcast domains: Each VLAN is a broadcast domain. This saves bandwidths and improves network throughput.
- Hardens LAN security: Packets from different VLANs are separately transmitted. Hosts in one VLAN cannot communicate with hosts in another without using IP addresses.
- Improves network robustness: A fault in one VLAN does not affect hosts in other VLANs.
- Flexibly defines virtual workgroups: Each VLAN defines a workgroup that can contain users from different geo-locations. This allows for flexible networking and easier maintenance.

A VLAN packet is a traditional Ethernet data frame with a four-byte 802.1Q tag inserted. This tag uniquely identifies a VLAN. Figure 3-3 and Figure 3-4 compare an ordinary data frame with a VLAN-tagged one.

Figure 3-3 Format of an ordinary Ethernet frame

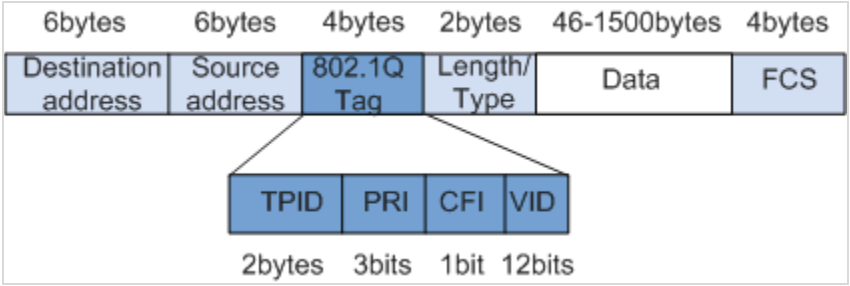


Figure 3-4 Format of a data frame with a VLAN tag

Each 802.1Q-capable switch sends data packets carrying a VLAN ID. The VLAN ID identifies the VLAN to which the switch belongs. Ethernet frames can be divided into the following two types depending on whether they are tagged or not:

- Tagged frame: Ethernet frame with a 4-byte 802.1Q tag.
- Untagged frame: original Ethernet frame without a 4-byte 802.1Q tag.

Although the OS and switch are both able to add a VLAN tag to data frames, VLAN tags are usually added or removed by a switch. Therefore, a VLAN has the following types of links:

- Access link: connects a host to a switch. Generally, the host does not need to know which VLAN it belongs to and the host hardware often cannot identify VLAN-tagged frames. Therefore, the host can only receive or send untagged frames.
- Trunk link: connects a switch to another switch or to a router. Data of different VLANs is transmitted along a trunk link. The two ends of a trunk link must be able to distinguish frames with different VLAN tags. Therefore, only tagged frames are transmitted along trunk links.

After the 802.1Q standard defines the VLAN frame, only certain interfaces of a device can identify a VLAN frame. Based on their ability to identify VLAN frames, interfaces are divided into two types:

- Access interface: The access interface resides on the switch and is used to connect the interface of the host. That is, the access interface can only connect the access link. Only packets carrying the default VLAN ID of this interface can pass through. Also, Ethernet frames sent through an access interface do not carry any tag.
- Trunk interface: A trunk interface is used to connect a switch to other switches and can connect only a trunk link. A trunk interface permits the tagged frames of multiple VLANs to pass through.

Interfaces of each type can be assigned with a default VLAN ID called PVID (Port Default VLAN ID). The meaning of a default VLAN varies with the interface type. The default VLAN ID of almost all switches is 1.

The following table lists the methods of processing data frames on various interfaces.

Table 3-1 Methods of processing data frames

Interface Type	Receiving Untagged Packet	Receiving Tagged Packet	Sending Frame
Access interface	Receive the packet, and tag it with the default VLAN ID.	If the VLAN ID of the packet is the same as the PVID of the interface, receive the packet. Otherwise, discard the packet.	Strip the PVID tag from the frame, and then send the frame.
Trunk interface	Tag the packet with the default VLAN ID. If the default VLAN ID is allowed on the interface, receive the packet. Tag the packet with the	If the VLAN ID is allowed on the interface, receive the packet. If the VLAN ID is not allowed on the interface, discard the	If the VLAN ID is the same as the default VLAN ID and is allowed on the interface, untag and then send the packet. If the VLAN ID is different from the default

Interface Type	Receiving Untagged Packet	Receiving Tagged Packet	Sending Frame
	default VLAN ID. If the default VLAN ID is not allowed on the interface, discard the packet.	packet.	VLAN ID and is allowed on the interface, keep the tag unchanged, and send the packet.

The following describes how a Huawei switch processes data frames.

- Configuration 1:

```
Port link-type access
Port default vlan 10
```

  In this configuration, the interface is an access interface and is configured with default VLAN 10. When a data frame with VLAN 10 arrives at the interface, its tag is removed and the untagged data frame is forwarded. When an untagged data frame arrives at the interface, it is tagged with VLAN 10 and is then forwarded. When a data frame with a tag different from VLAN 10 arrives at the interface, it is discarded.

- Configuration 2:

```
Port link-type trunk
Port trunk pvid 10
Port trunk allow-pass vlan 16 17
```

  In this configuration, the interface is a trunk interface and is configured with default VLAN 10. This interface allows packets with VLAN 16 or VLAN 17 to pass through. When an untagged data frame arrives at the interface, it is tagged with VLAN 10. When a data frame with VLAN 10 arrives at the interface, its tag is removed and the untagged data frame is forwarded. When a data frame with VLAN 16 or VLAN 17 arrives at the interface, it is directly forwarded. When any other data frame arrives at the interface, it is discarded.

VLANs can be divided based on switch ports, MAC addresses, subnets, or policies. A VLAN can be further divided into multiple VLANs, which is an advanced VLAN function. For details, see Huawei data communication certification courses.

3.2 Physical Networks in Virtualization

In virtualization, applications run on VMs, and VMs run on physical servers. Before connecting VMs to a network, connect physical servers to the network first. To do so, the following devices are required: routers, layer-3 switches, layer-2 switches, and server NICs.

Before discussing physical devices, we will learn about the four-layer TCP/IP model.

TCP/IP is the most widely used protocol stack used on the Internet. TCP/IP divides the entire communication network into four layers, namely, the application layer, transport layer, network layer, and link layer, as shown in the following figure.

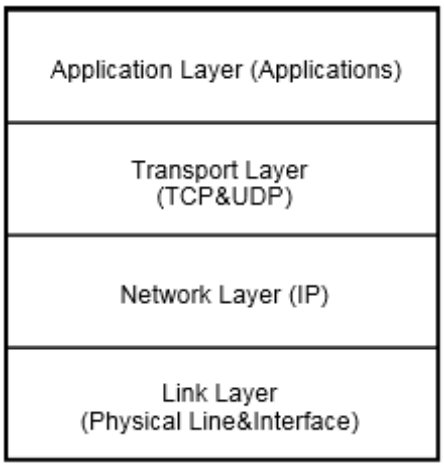


Figure 3-5 TCP/IP protocol stack

The routing process works at the transport layer (layer 3), and VLAN works at the network layer (layer 2). Therefore, if a device has the routing function and can look up the routing table, it is considered a layer-3 device. If a device supports VLAN configuration only, it is considered a layer-2 device. A hub is a link-layer (layer 1) device although it appears to function as a switch does. The reason is that the hub is only like a splitter and does not support the function of dividing VLANs. Routers and layer-3 switches work at the transport layer, layer-2 switches work at layer 2, and the network cables and optical fibers of physical server NICs and link NICs work at layer 1.

Both routers and layer-3 switches work at the transport layer. Layer-3 switches have the routing function, and many routers configured with an additional switching board support the function of dividing VLANs. Then, the question is whether a layer-3 switch and a router can substitute each other. The answer is no.

First, a switch and a router provide different functions. A switch uses the dedicated chip (ASIC) for high-speed data exchange while a router maintains a route table for route addressing across IP address ranges. A router has the nature of separating broadcast domains. Even if a router gains the switching function or a switch obtains the routing function, its key function remains unchanged, with the new function only as a supplement.

Second, switches are mainly used in LANs, but routers are mainly used in WANs. The LAN features frequent data exchanges, a single network interface type, and a large number of interfaces. The switch can provide fast data forwarding, the network cable interface (RJ45), and the optical fiber interface. In addition, a switch can provide a large number of interfaces to meet the requirements of the LAN. The WAN features a variety of network types and a variety of interface types. The router provides a powerful routing function, which can be used not only between LANs using one protocol but also between LANs and WANs using multiple protocols. The WAN has the following advantages: selecting the optimal route, balancing load, backing up links, and exchanging route information with other networks. These are functions provided by the router.

Third, the performance of the layer-3 switch is different from that of the router. Technically speaking, the router and the layer-3 switch have distinct differences in data packet switching. The router uses the software engine with a micro-processor to forward data packets while the layer-3 switch uses hardware. After a layer-3 switch forwards the first packet of a data flow, it generates a mapping between MAC addresses and IP addresses. When the same data flow passes, the layer-3 switch forwards the packets without routing again. This prevents the delay caused by route selection and improves the efficiency of forwarding data packets.

Copyright © Huawei Technologies Co., Ltd.

In addition, the layer-3 switch searches for routes for data flow. The layer-3 switch uses the cache technology to achieve the function with ease, lowering costs dramatically and implementing fast forwarding. The router uses the longest-match mode to forward packets. This complex process is usually implemented by software, with a low forwarding efficiency. Therefore, in terms of performance, the layer-3 switch is better than the router and is applied to the LAN with frequent data exchange. With a powerful routing function and low forwarding efficiency of data packets, the router is applied to the connection of different types of networks without frequent data exchange, such as the connection between the LAN and Internet. If the router, especially the high-end router, is used on a LAN, its powerful routing function is wasted and it cannot meet the communication requirements of the LAN and adversely affects subnet communication.

In cloud computing and virtualization, routers are usually deployed at the egress of an enterprise or institution to connect the intranet to the Internet. When an intranet user intends to access the Internet, the router will perform route forwarding and network address translation (NAT). When an Internet user intends to access the intranet, the traffic will also pass through the router.

The router can allow the intranet to communicate with the Internet, which means that one issue is addressed. The other issue is how to plan the intranet. Our purpose is to connect physical servers to the network. In the equipment room of a data center, servers are placed in racks. For better cooling, racks are arranged in rows. Therefore, two popular ways for connecting servers to a network are available, which are Top of Rack (ToR) and End of Row (EoR). ToR and EoR are terms defining cabling in a rack. ToR means to place the switch for connecting servers to the network on the top of racks. If servers are densely deployed in racks and bear massive traffic, one ToR switch will be placed on the top of each rack. If servers are loosely deployed in racks and bear average traffic, one ToR switch may be shared by multiple racks. When selecting a ToR switch, select a GE or 10 GE one based on your needs. The ToR switch is used to connect network ports of physical servers to the aggregation switch or the core switch, connecting the physical servers to the network. The following figure shows the ToR cabling mode.

Figure 3-6 ToR cabling mode

EoR differs from ToR in that an independent rack among a row of racks is used to deploy the switch for connecting servers to a network. This switch is called an EoR switch. As its name implies, End of Row (EoR) reveals that the switch is placed at the end. However, in many cases, the switch is placed in the middle of a rack row to reduce the length of cables required between servers and the switch. Network cable distribution frames and optical fiber

distribution frames are prepared in advance for other racks. Network ports of servers are directly connected to distribution frames, and are then connected to the EoR switch. The following figure shows the details.

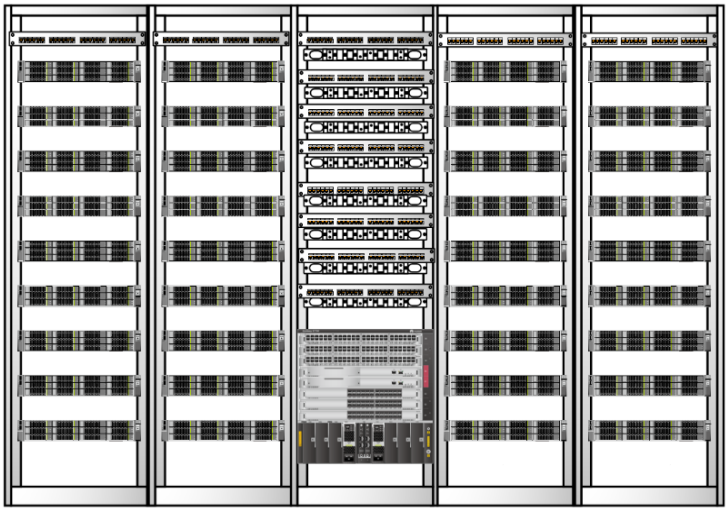


Figure 3-7 EoR cabling mode

ToR and EoR are suitable for their own scenarios and have limitations.

EoR limitations: You have to lay out cables between the server rack and the network rack in advance. Therefore, servers in the equipment room must be planned and deployed elaborately. For example, you have to determine the number of servers, the number of network ports on each server, and available types of network port. The farther the network rack is apart from the server rack, the longer the required cable in the equipment room is. In addition, to ensure a tidy environment in the equipment room, cables must be bundled in order. When the quantity of required cables is different from the quantity of existing cables, or when a cable fault occurs, the cables must be laid out again. As a result, heavy cable management and maintenance workloads are generated, with low flexibility.

ToR limitations: Each server rack has a fixed output power. Therefore, only a limited number of servers can be deployed, and only 12 to 16 ports are available. Currently, a switch usually has at least 24 ports. As a result, the access port usage of switches in the rack is low. If multiple server racks share one or two access switches, the switch port usage will no longer be low. However, this is like small-scale EoR cabling, which results in more cable management and maintenance.

When drawing an overall network cabling plan, work out an optimal solution by considering the respective advantages and limitations of EoR and ToR.

After servers are connected to the network, they are divided by the type of network traffic, which can be service traffic, management traffic, or storage traffic. Service traffic and storage traffic are vital to users. When users access a target service, service traffic is produced. If service data is stored on a professional storage device instead of a local server, storage traffic is produced when a server accesses the storage device. When users manage servers, virtualization devices, and storage devices, management traffic is produced. Currently, nearly every physical device is configured with an independent management interface. If management traffic and service traffic are carried on different physical lines and interfaces, this is out-of-band management. If management traffic and service traffic are carried on a same physical channel, this is in-band management.

In a cloud computing data center, a high-end layer-3 switch is used as the core of the entire network during network design. The default gateways of all traffic-related network segments are configured on the switch. In this way, all inter-broadcast-domain access will pass through the switch. The reasons for this measure are as follows:

- The high-end layer-3 switch has a high forwarding performance and can meet the requirements for forwarding traffic on the entire network.
- The high-end layer-3 switch has a modular structure, with excellent fault tolerance and high scalability. The high-end layer-3 switch has such necessary modules as the power supply and the fan, and has also a core component, which is the engine board. They are all deployed in 1+1 hot backup, improving device availability. The switching board is hot swappable, enabling users to scale out network resources at any time.
- The high-end layer-3 switch provides boards with various interface densities, such as 10GE, 40GE, and 100GE. The high-end layer-3 switch supports large-capacity, high-density server access and ToR aggregation.
- Apart from such basic functions as routing and switching, the high-end layer-3 switch supports other features that suit cloud computing, such as large layer 2, stacking, and virtualization.

All traffic is first transmitted to the layer-2 switch before being transmitted to the core switch. The access modes are EoR and ToR, which we have discussed earlier. Based on the types of incoming traffic, access switches are divided into management, storage, and service switches. For a data center with a huge traffic volume, it is recommended that a specific physical switch be used to carry a specified type of traffic. This means that the switch for each type of traffic is an independent device. For a data center with an average traffic volume, a same physical switch can be used to handle several different types of traffic, and traffic flows are logically isolated from each other using VLANs. The following figure shows the situation where all traffic is physically and logically isolated.

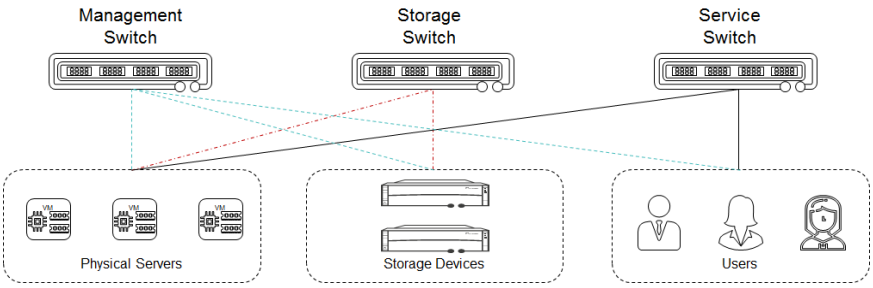


Figure 3-8 Access layer switch

A physical server uses its own physical NIC to connect to the network. All VM traffic enters the entire network through various types of network ports. The physical NIC involves a key concept, which is port (link) aggregation. Port aggregation indicates that multiple physical Ethernet links are bonded into one logical link to increase the link bandwidth. In addition, the bundled links dynamically back up each other, greatly improving link reliability.

As the network scale expands, users have increasingly high requirements on the bandwidth and reliability of the backbone link. Originally, to increase the bandwidth, users used high-speed devices to replace old devices. This solution, however, is costly and inflexible.

LACP allows multiple physical ports to be bonded into a logical port to increase the link bandwidth without upgrading hardware. In addition, the link backup mechanism of LACP provides higher link transmission reliability.

LACP has the following advantages:

- Increased bandwidth

  The maximum bandwidth of a link aggregation port reaches the total bandwidth of all member ports.
- Higher reliability

  If an active link fails, the traffic is switched to another available member link, improving reliability of the link aggregation port.
- Load balancing

  In a link aggregation group, loads can be shared among member active links.

Link aggregation can work in manual load balancing mode and LACP mode.

In manual load balancing mode, users must manually create a link and add member interfaces into the link. In this case, the LACP protocol is not involved. In this mode, all active links work in load-sharing mode to forward data. If an active link is faulty, the remaining active links evenly share the traffic. If a high link bandwidth between two directly connected devices is required but one of the devices does not support the LACP protocol, you can use the manual load balancing mode.

The manual load balancing mode cannot detect other faults, such as link layer faults and incorrect link connections. To improve link fault tolerance and provide the backup function for high reliability of member links, the LACP protocol is introduced. The LACP mode is actually a link aggregation mode using LACP. LACP provides a standard negotiation mode for devices that exchange data, so that devices can automatically form an aggregated link and start this link to send and receive data according to its own configuration. After the aggregation link is set up, LACP maintains the link status and implements dynamic link aggregation and deaggregation.

3.3 Virtual Networks in Virtualization

3.3.1 Virtual Network Architecture

As cloud computing and virtualization are becoming increasingly popular, layer-2 access switches no longer function as the network access layer. Instead, layer-2 access switches will be deployed on servers to connect VMs. In this case, virtual switches are required and they function as a real network access layer.

In previous sections we have discussed about the advantages of cloud computing and virtualization. Due to these advantages, they have now become the mainstream IT technologies. However, with every new technology comes new challenges. With the widespread use of virtualization, it is no longer the physical server that runs the workloads. With a physical server, a dedicated network cable is used to connect to a switch, and the few workloads running on this server share this network cable. Currently, multiple VMs run on each physical server and use one network cable to transmit various types of traffic. The new challenge now is how to manage different types of traffic and their statuses.

Traditionally, the IT staff is divided into host engineers, network engineers, and software developers. Each of these roles has clear-cut responsibilities. In a cloud or virtualized environment, however, virtual switches run inside servers, so that when a fault occurs in such an environment, it is not always easy to determine who gets to troubleshoot the virtual switches: the network engineer or the host engineer. Therefore, it is necessary for both the network engineer and the host engineer to acquaint them with the architecture of the virtual network.

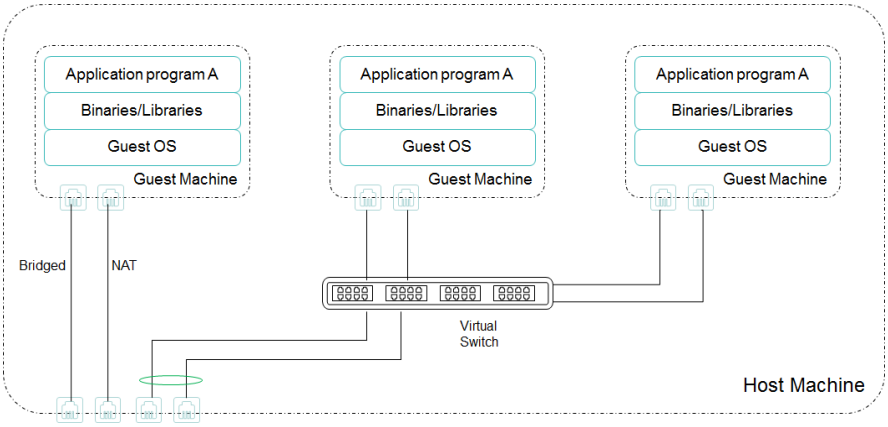


Figure 3-9 Architecture of a virtual network

A common virtualization system uses the architecture shown in the preceding figure. In a personal or small-scale virtualization system, VMs are bound to physical NICs using bridges or NAT. In a large-scale corporate virtualization system, VMs are connected to physical networks using virtual switches.

Network bridge is not a new technology. You can use network bridges to enable network ports to communicate with each other. Specifically, you can use network bridges to interconnect multiple network ports so that packets received on one network port will be replicated to the others.

In a virtualization system, the OS is responsible for interconnecting all network ports. The following figure shows bridge-based interconnection in a Linux system.

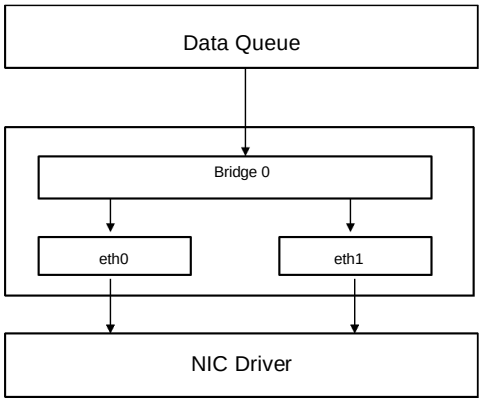


Figure 3-10 Bridge-based interconnection in a Linux system

Bridge-based interconnection in a Linux system

Bridge 0 (network bridge device) is bound to **eth0** and **eth1**. The upper-layer network protocol stack knows only **Bridge 0** and does not need to know bridging details because the bridge is implemented at the data link layer. When the upper-layer network protocol stack needs to send packets, it sends the packets to **Bridge 0**, and the processing code of **Bridge 0** determines whether the packets will be forwarded to **eth0**, **eth1**, or both. Similarly, packets received on **eth0** or **eth1** are sent to **Bridge 0**, and the processing code of **Bridge 0** determines whether to forward the packets, discard the packets, or send the packets to the upper-layer network protocol stack.

The network bridge provides the following key functions:

- MAC learning

  The network bridge learns MAC addresses. Initially, the network bridge has no mapping between MAC addresses and ports and sends data like a HUB. However, when sending data, the network bridge learns the MAC addresses of data packets and finds the corresponding ports, setting up a mapping between MAC addresses and ports (CAM table).

- Packet forwarding

  When sending each data packet, the network bridge obtains its destination MAC address and searches the MAC address-port table (CAM table) to determine the port through which the data packet will be sent.

With virtualization, each VM has a virtual NIC. A Linux OS will generate a TAP device in user mode and a tun/tap device driver and NIC driver in kernel mode. Data sent from a VM passes through the tun/tap character device file in user mode, passes through the tun/tap device driver and virtual NIC driver in kernel mode, is sent to the TCP/IP protocol stack, and is then forwarded to the virtual NIC tap in user mode. The tap is now directly connected to the network bridge, and **eth0** and **eth1** in the preceding figure will act as the NIC tap of the VM. The following figure describes the details.

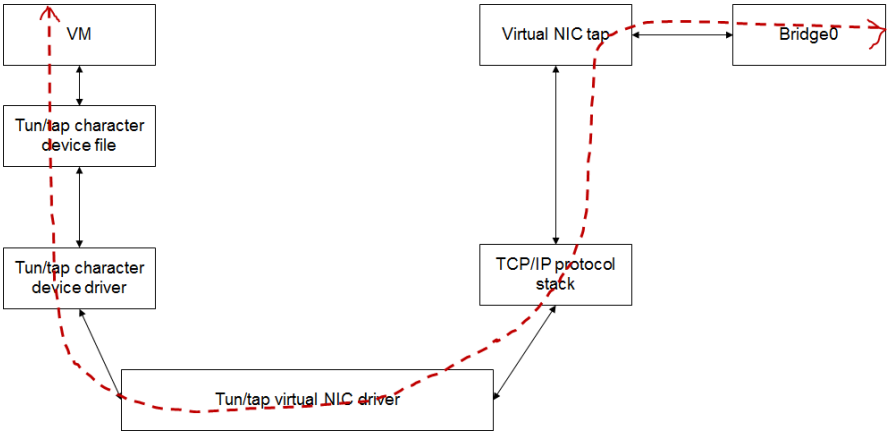


Figure 3-11 Forwarding traffic over a network bridge

NOTE

Log in to CNA and run the **ifconfig** command to view tap devices. Two tap configuration files are displayed, as shown in the following figure.

```
tap00000001.0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether fe:6e:d4:88:b5:06  txqueuelen 5000  (Ethernet)
        RX packets 2633199  bytes 662047312 (631.3 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3682209  bytes 1012247487 (965.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

tap00000002.0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether fe:6e:d4:88:c6:29  txqueuelen 5000  (Ethernet)
        RX packets 9  bytes 2102 (2.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 35  bytes 2202 (2.1 KiB)
        TX errors 0  dropped 514913 overruns 0  carrier 0  collisions 0
```

The information reveals that the MAC addresses of the two tap devices are **fe:6e:d4:88:b5:06** and **fe:6e:d4:88:c6:29**. Log in to the VRM page and query the MAC addresses of the existing

VMs. You will find that the MAC addresses of the two tap devices match the MAC addresses of the existing VMs.

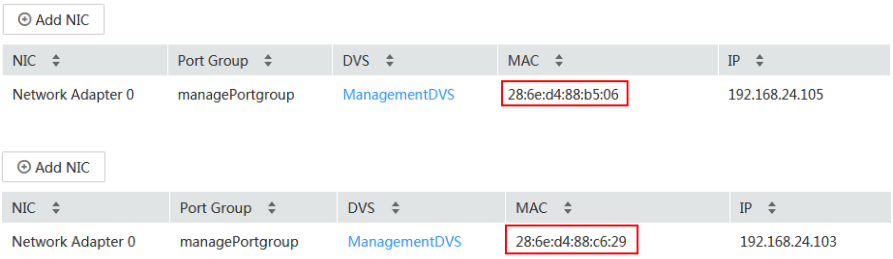


Figure 3-12 MAC addresses of the existing VMs

If only the network bridge is used, VMs can communicate with external networks using bridges or NAT. When a bridge is used, the network bridge functions as a switch, and the virtual NIC is connected to a port of the switch. If NAT is used, the network bridge functions as a router, and the virtual NIC is connected to a port of the router.

When the virtual NIC is connected to a port of the switch, the virtual NIC and the network bridge, with the same IP address configuration, communicate with each other in broadcast mode. When the virtual NIC is connected to a port of the router, the virtual NIC and the network bridge are configured with IP addresses that belong to different IP address ranges. In this case, the system automatically generates an IP address range, the virtual NIC communicates with other networks including the network bridge in layer-3 routing forwarding mode, and address translation is conducted on the network bridge. In the Linux system, the IP address range generated by default is **192.168.122.0/24**, as shown in the following figure.

```
virbr0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.122.1  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 52:54:00:cb:7e:97  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

vnet0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::fc54:ff:feeb:e118  prefixlen 64  scopeid 0x20<link>
        ether fe:54:00:eb:e1:18  txqueuelen 1000  (Ethernet)
        RX packets 685272  bytes 74578135 (71.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1232006  bytes 1260248954 (1.1 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 3-13 IP address of the network bridge using NAT in the Linux system

NAT is used for address translation. With NAT used for address translation, when a VM communicates with an external network through the NAT gateway, which is **virbr0** in the preceding figure, the source IP address of the IP packet is translated into the IP address of the physical network bridge and a record is produced accordingly. When the external network accesses the VM, the NAT gateway forwards data packets to the VM based on the record.

NAT is widely used and has the following advantages:

- When the IP address range for the physical network bridge has insufficient available IP addresses, a new IP address range can be added.

- The source IP address can be concealed. When a VM accesses an external network, the IP address of the VM is translated on the NAT gateway. Therefore, the external network will not know the IP address, protecting VM security.
- Load balancing NAT provides the redirecting function. When multiple VMs with the same applications are deployed in active/standby mode, NAT can translate their IP addresses into one IP address used for communication with external networks. In addition, load balancing software is used for evenly distributing service access.

The bridge and NAT are suitable for personal or small-scale systems. When the bridge is used, the statuses of virtual NICs cannot be viewed and the traffic on virtual NICs cannot be monitored. The bridge supports only the GRE tunnel, with limited functions. The network bridge does not support software-defined networking (SDN), which is widely used currently. Therefore, in a large-scale system, the virtual switch is used for VMs to communicate with external networks. The virtual switch acts as an upgraded network bridge, removing the defects of the network bridge.

Currently, each virtualization vendor has its own virtual switching product, such as VMware vSwitch, Cisco Nexus 1000V, and Huawei DVS. The following describes open-source Open vSwitch.

Open vSwitch (OVS) is an open-source, high-quality, and multi-protocol virtual switch. It is developed by Nicira Networks using the open-source Apache2.0 license protocol. Its main codes are portable C codes. Open vSwitch is designed to enable massive network automation through programmatic extension and also support the standard management interfaces and protocols, such as NetFlow, sFlow, SPAN, RSPAN, CLI, LACP, and 802.1ag. Open vSwitch supports multiple Linux virtualization technologies, such as Xen and KVM.

The OVS official website describes Why-OVS as follows:

- The mobility of state

  All network state associated with a network entity (say a virtual machine) should be easily identifiable and migratable between different hosts. This may include traditional "soft state" (such as an entry in an L2 learning table), L3 forwarding state, policy routing state, ACLs, QoS policy, monitoring configuration (e.g. NetFlow, IPFIX, sFlow), etc. Open vSwitch has support for both configuring and migrating both slow (configuration) and fast network state between instances. For example, if a VM migrates between end-hosts, it is possible to not only migrate associated configuration (SPAN rules, ACLs, QoS) but any live network state (including, for example, existing state which may be difficult to reconstruct). Further, Open vSwitch state is typed and backed by a real data-model allowing for the development of structured automation systems.
- Responding to network dynamics

  Virtual environments are often characterized by high-rates of change. VMs coming and going, VMs moving backwards and forwards in time, changes to the logical network environments, and so forth. Open vSwitch supports a number of features that allow a network control system to respond and adapt as the environment changes. This includes simple accounting and visibility support such as NetFlow, IPFIX, and sFlow. But perhaps more useful, Open vSwitch supports a network state database (OVSDB) that supports remote triggers. Therefore, a piece of orchestration software can "watch" various aspects of the network and respond if/when they change. This is used heavily today, for example, to respond to and track VM migrations. Open vSwitch also supports OpenFlow as a method of exporting remote access to control traffic. There are a number of uses for this including global network discovery through inspection of discovery or link-state traffic (e.g. LLDP, CDP, OSPF, etc.).
- Maintenance of logical tags

Distributed virtual switches often maintain logical context within the network through appending or manipulating tags in network packets. This can be used to uniquely identify a VM (in a manner resistant to hardware spoofing), or to hold some other context that is only relevant in the logical domain. Much of the problem of building a distributed virtual switch is to efficiently and correctly manage these tags. Open vSwitch includes multiple methods for specifying and maintaining tagging rules, all of which are accessible to a remote process for orchestration. Further, in many cases these tagging rules are stored in an optimized form so they don't have to be coupled with a heavyweight network device. This allows, for example, thousands of tagging or address remapping rules to be configured, changed, and migrated. In a similar vein, Open vSwitch supports a GRE implementation that can handle thousands of simultaneous GRE tunnels and supports remote configuration for tunnel creation, configuration, and tear-down. This, for example, can be used to connect private VM networks in different data centers.

NOTE

GRE provides a mechanism for encapsulating packets of one protocol into packets of another protocol. For example, OSPF is used on the LAN, EGP is used on the WAN, and GRE is used to encapsulate OSPF packets into EGP packets before they are transmitted. When you are mailing an international letter, the address on the envelope may be written in English while the letter itself may be written in your mother tongue. In this case, you mother tongue is like OSPF, the English language is like EGP, and the envelope wrapping the letter is like the GRE technology.

- Hardware integration

  Open vSwitch's forwarding path (the in-kernel datapath) is designed to be amenable to "offloading" packet processing to hardware chipsets, whether housed in a classic hardware switch chassis or in an end-host NIC. This allows for the Open vSwitch control path to be able to both control a pure software implementation or a hardware switch. There are many ongoing efforts to port Open vSwitch to hardware chipsets. These include multiple merchant silicon chipsets (Broadcom and Marvell), as well as a number of vendor-specific platforms. The advantage of hardware integration is not only performance within virtualized environments. If physical switches also expose the Open vSwitch control abstractions, both bare-metal and virtualized hosting environments can be managed using the same mechanism for automated network control.

In summary, in many ways, Open vSwitch targets a different point in the design space than previous hypervisor networking stacks, focusing on the need for automated and dynamic network control in large-scale Linux-based virtualization environments.

The goal with Open vSwitch is to keep the in-kernel code as small as possible (as is necessary for performance) and to re-use existing subsystems when applicable (for example Open vSwitch uses the existing QoS stack). As of Linux 3.3, Open vSwitch is included as a part of the kernel and packaging for the userspace utilities are available on most popular distributions.

OVS has the following key components:

- ovs-vswitchd: This is the main module of OVS, which is used to implement the daemon of the switch and which includes a Linux kernel module that supports stream switching.
- ovsdb-server: This is a lightweight database server that enables ovs-vswitchd to obtain configuration information.
- ovs-dpctl: This is used to configure the kernel module of the switch.
- Some auxiliary OVSs with scripts and specifications are installed on Citrix XenServer as default switches.
- ovs-vsctl: This is used to query and update the ovs-vswitchd configuration.
- ovs-appctl: This is used to send command messages and run related daemons.

In addition, OVS provides the following tools:

- ovs-ofctl: This is used to query and control the OpenFlow switch and controller.
- ovs-pki: This is the public key framework used to create and manage OpenFlow switches.

The following figure shows the process for forwarding data packets on OVS.

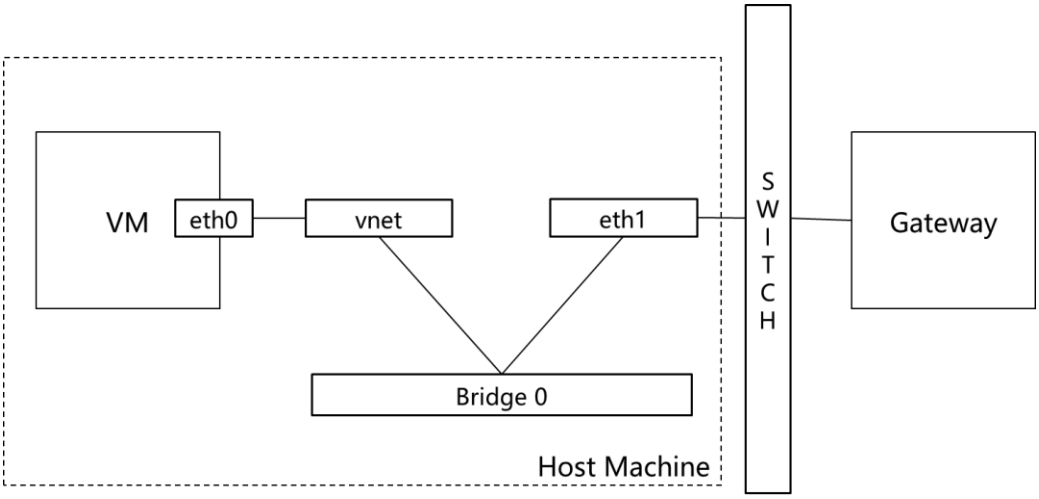


Figure 3-14 Forwarding data packets on OVS

1. The data packets generated on the VM are sent to the virtual NIC **eth0**.
2. The data packets are first sent to the tun/tap device, which is **vnet** in the preceding figure.
3. They are then transmitted by **vnet** to the network bridge.
4. Finally, the data packets are forwarded by the network bridge to the physical machine NIC **eth1** that is connected to the network bridge, and the data packets are forwarded by **eth1** to the physical layer-2 switch.

The virtual switch can be a common virtual switch or a distributed virtual switch (DVS). A common virtual switch runs only on a single physical host. All network configurations apply only to VMs on the physical host. Distributed virtual switches are deployed on different physical hosts. You can use the virtualization management tool to configure distributed virtual switches in a unified manner. The distributed virtual switch is required for VM live migration.

Huawei virtualization products use distributed virtual switches. This section describes Huawei distributed virtual switches.

3.3.2 Network Features of Huawei Virtualization Products

3.3.2.1 Network Solutions in Huawei Virtualization Products

Huawei distributed virtual switches can be centrally managed. The centralized management modules provide a unified portal for simplified configuration management and user management.

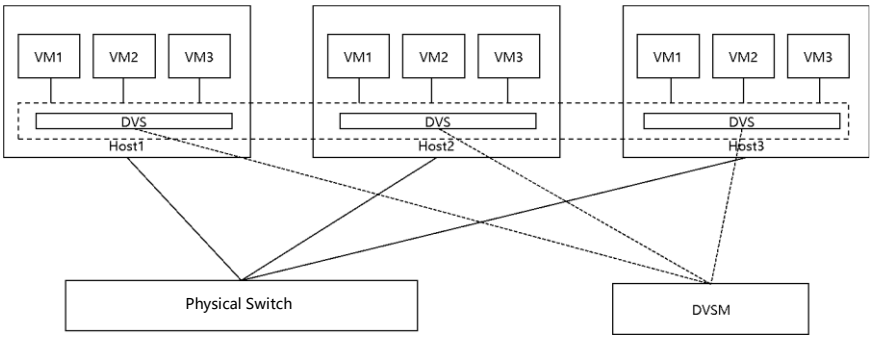


Figure 3-15 DVS

The virtual switches distributed on physical servers provide VMs with a range of capabilities, including layer-2 connectivity, isolation, and QoS.

The DVS model has the following characteristics:

- Multiple DVSs can be configured, and each DVS can serve multiple CNA nodes in a cluster.
- A DVS provides several virtual switch ports (VSP) with configurable attributes, such as the rate and statistics. Ports with the same attributes are assigned to the same port group for easy management. Ports that belong to the same port group are assigned the same VLAN.
- Different physical ports can be configured for the management plane, storage plane, and service plane. An uplink port or an uplink port aggregation group can be configured for each DVS to enable external communication of VMs served by the DVS. An uplink aggregation group comprises multiple physical NICs working based on preconfigured load-balancing policies.
- Each VM provides multiple virtual NIC (vNIC) ports, which connect to VSPs of the switch in one-to-one mapping.
- A server allowing layer-2 migration in a cluster can be specified to create a virtual layer-2 network based on service requirements and configure the VLAN used by this network.

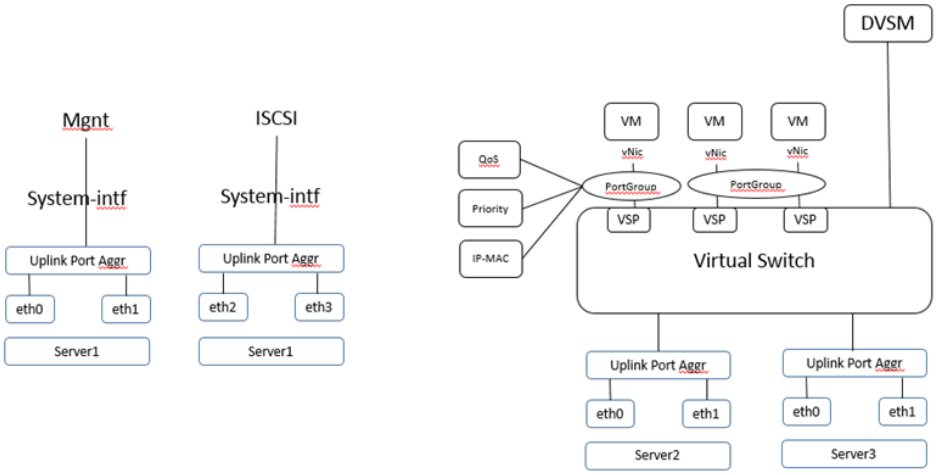


Figure 3-16 Internal structure of a virtual switch

The configuration of VM port attributes can be simplified by configuring port group attributes instead, including security and QoS. Modifying port group attributes has no impact on the proper running of VMs.

A port group consists of multiple ports with the same attributes. By configuring the port group attributes, including bandwidth QoS, layer-2 security attributes, and VLAN, the administrator configures the attributes of all VM ports in the group, thus saving a lot time. Port group attribute changes do not affect VM running.

An uplink connects the host and the DVS. Administrators can query information about an uplink, including its name, rate, mode, and status.

Uplink aggregation allows multiple physical ports on a server to be bound as one port to connect to VMs. Administrators can set the bound port to load balancing mode or active/standby mode.

Huawei DVS supports the virtual switching function of open-source Open vSwitch. The following figure shows the structure of Huawei DVS.

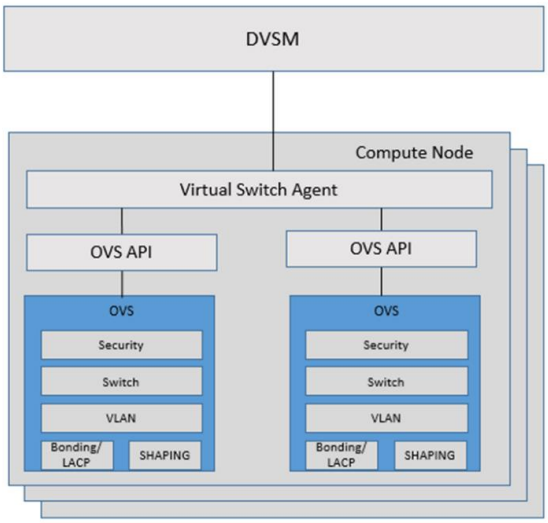


Figure 3-17 Internal structure of Huawei DVS

Huawei DVS has the following characteristics:

- Unified portal and centralized management simplify user management and configuration.
- The open-source Open vSwitch is integrated to make full use of and inherit virtual switching capabilities of open source communities.
- Rich virtual switching layer-2 features, including switching, QoS, and security isolation, are provided.

3.3.2.2 DVS Traffic Flow

- VMs run on the same host but belong to different port groups.

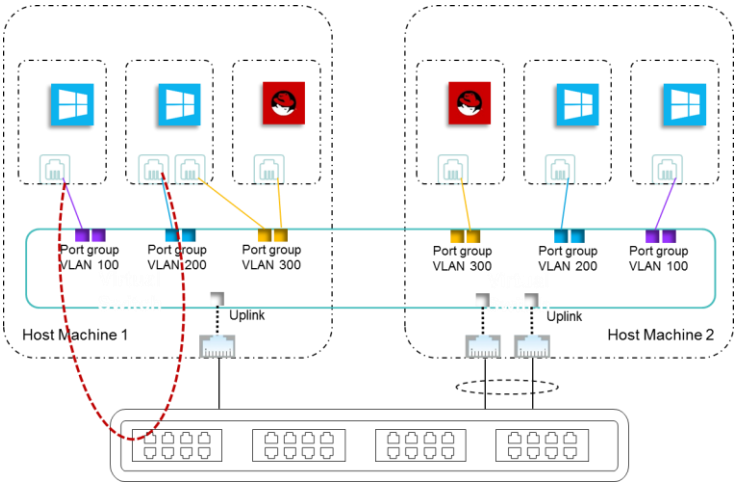


Figure 3-18 Traffic flow when VMs run on the same host but belong to different port groups

A virtual switch is essentially a layer-2 switch. The port group has a vital parameter, which is VLAN ID. If two VMs belong to different port groups, they are associated with different VLANs, and cannot detect each other using broadcast. Generally, when two VMs are associated with different VLANs, they are configured with IP addresses that belong to different IP address ranges. Enabling communication between the two VMs requires a layer-3 device, which can be a layer-3 switch or router. In Huawei FusionCompute, layer-3 switching can only be performed by physical layer-3 devices. Therefore, when these two VMs intend to communicate with each other, the traffic needs to come from one host and arrive at the physical access switch, and then be forwarded to the peer layer-3 device and routed to the other host.

- VMs run on the same host and belong to the same port group.

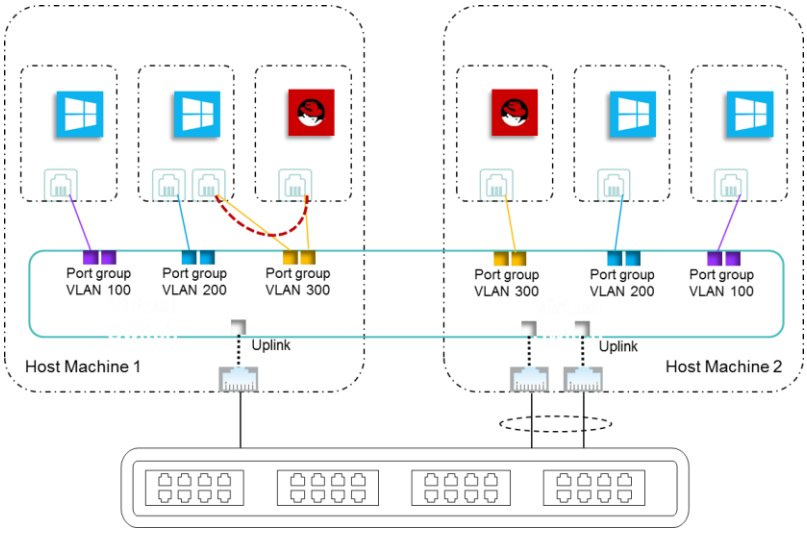


Figure 3-19 Traffic flow when VMs run on the same host and belong to the same port group

When two VMs run on the same host and belong to the same port group, they belong to the same broadcast domain, in which case, they can communicate with each other through a virtual switch and the traffic will not enter the physical network.

- VMs run on different hosts but belong to the same port group.

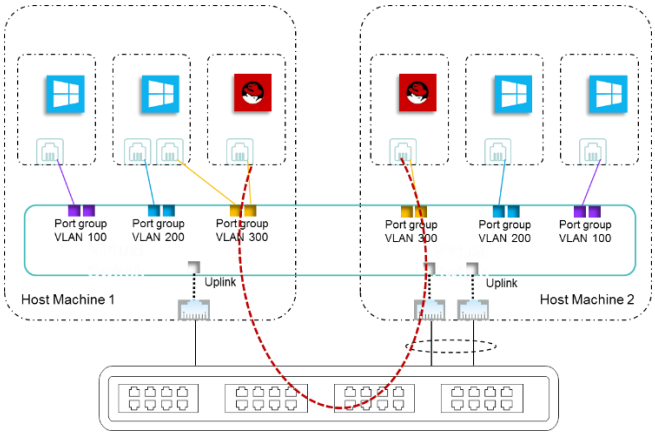


Figure 3-20 Traffic flow when VMs run on different hosts but belong to the same port group

When two VMs belong to the same port group, they may detect each other by sending out broadcast packets. However, because the two VMs run on different hosts, they need a physical switch to connect to the network. An exception is that the two physical servers are directly connected. Therefore, when two VMs that run on different hosts but belong to the same port group intend to communicate with each other, the traffic needs to pass through the physical switch. The two VMs can communicate with each other without using the layer-3 device, which is different from the situation when the two VMs run on different physical servers and belong to different port groups.

- Multiple DVSs run on a physical host.

  In real-world scenarios, multiple DVSs usually run on the same physical host. Generally, when two VMs are connected to different DVSs, the port groups associated with the two DVSs have different VLAN IDs, which means that the two VMs use different IP addresses. In this case, the traffic between the two VMs will need to be routed through a layer-3 device.

3.3.2.3 Security Group

Users can create security groups based on VM security requirements. Each security group provides a set of access rules. VMs that are added to a security group are protected by the access rules of the security group. Users can add VMs to security groups for security isolation and access control when creating VMs. A security group is a logical group that consists of instances that have the same security protection requirements and trust each other in the same region. All VM NICs in a security group communicate with each other by complying with the security group rules. A VM NIC can be added to only one security group.

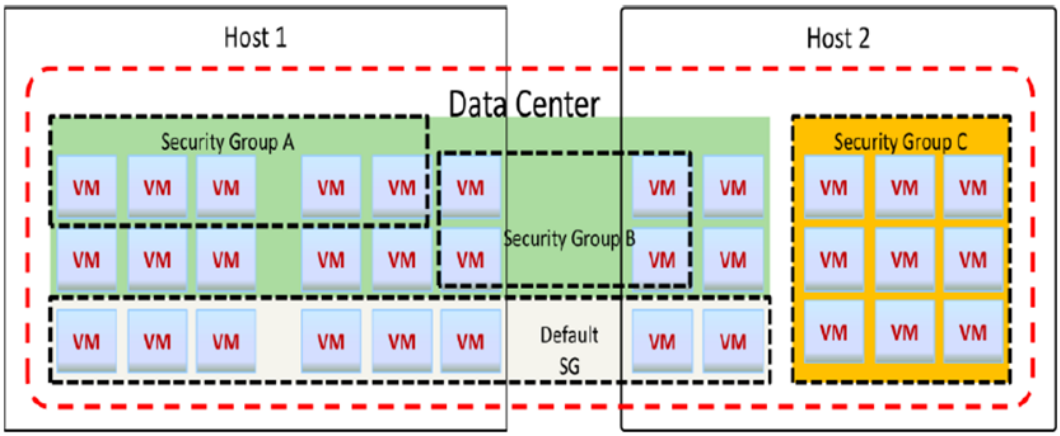


Figure 3-21 Security group

The security group provides a similar function as the firewall does. They both use iptables to filter packets for access control.

Netfilter/iptables (iptables for short) functions as the firewall filtering packets on the Linux platform. iptables is a Linux userspace module located in **/sbin/iptables**. Users can use iptables to manage firewall rules. netfilter is a Linux kernel module that implements the firewall for packet filtering.

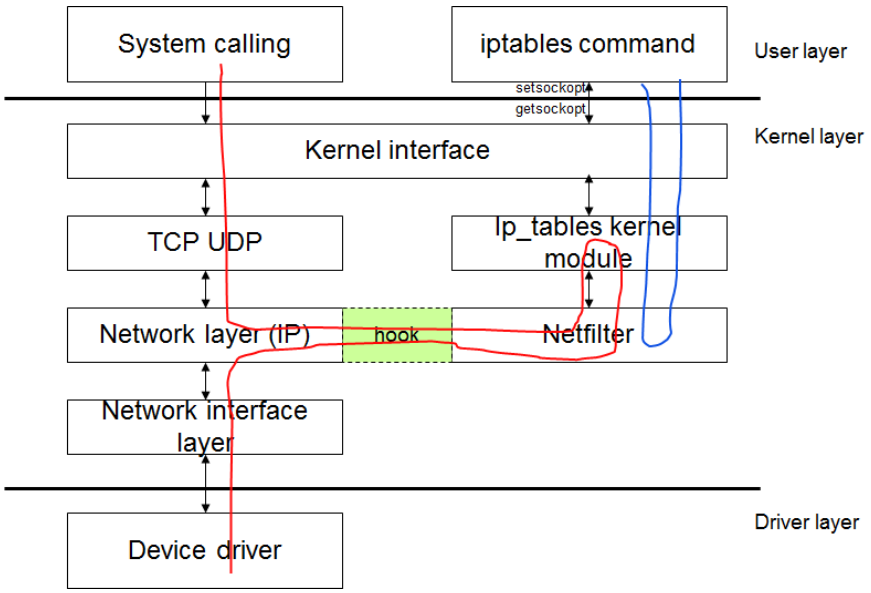


Figure 3-22 Working principles of the security group

In the preceding figure, Netfilter is a data packet filtering framework. When processing IP data packets, it hooks five key points, where services can mount their own processing functions to implement various features.

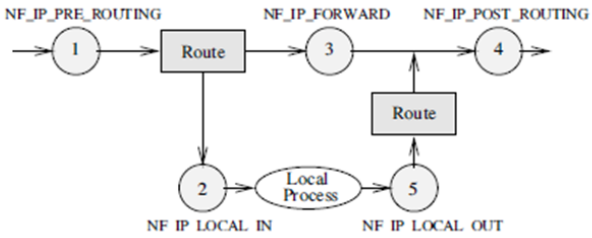


Figure 3-23 Filtering packets using rules

The iptables configuration is processed based on chains and rules in the table. Network packets that enter the chain match the rules in the chain in sequence. When a packet matches a rule, the packet is processed based on the action specified in the rule. iptables contains four tables, which are RAW, Filter (filtering packets), NAT (translating network addresses), and Mangle (changing TCP headers). These tables generate their own processing chains at the five key points as required and mount the entry functions for chain processing to the corresponding key point.

3.3.2.4 Virtual Switching Mode

The Huawei virtual switch provides the following virtual switching modes: common mode, SR-IOV mode, and user mode.

- Common mode

  In this mode, a VM has two vNICs, front-end vNIC and back-end vNIC. The front-end vNIC connects to a virtual port of the virtual switch. VM network packets are transmitted between the front- and back-end vNICs through an annular buffer and event channel, and forwarded by the virtual switch connected to the back-end vNIC. The following figure shows the details.

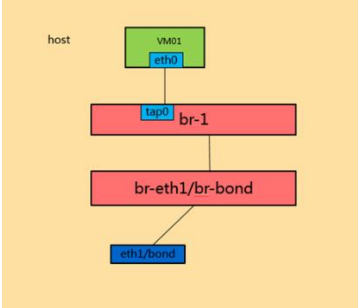


Figure 3-24 Common mode

- SR-IOV mode

  Single Root I/O Virtualization (SR-IOV) is a network I/O virtualization technology proposed by Intel in 2007 and is now a PCI-SIG standard. A physical NIC that supports SR-IOV can be virtualized into multiple NICs for VMs and it seems that the VMs enjoy an independent physical NIC. This improves network I/O performance compared with software virtualization and this requires fewer hardware NICs compared with PCI Passthrough. The following figure shows the details.

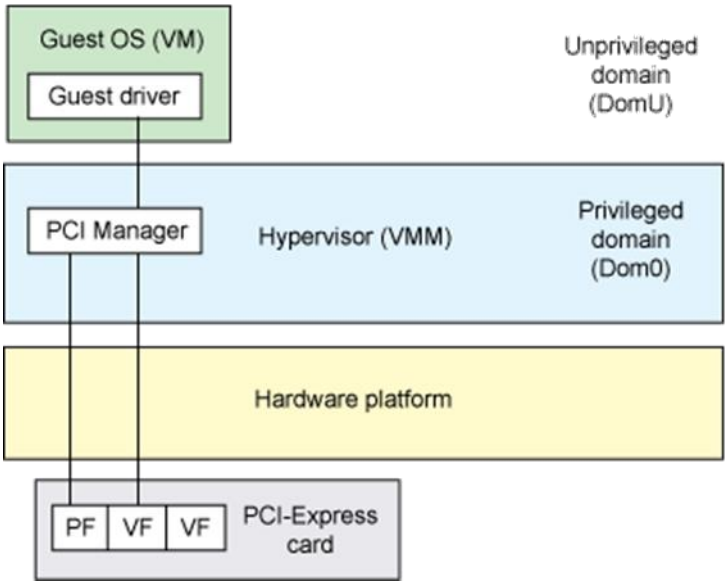


Figure 3-25 SR-IOV mode

- User mode

  The user-mode driver is loaded to the virtual port, and a thread is started in vswitchd to replace the function of receiving and sending packets in a kernel state. A data packet received from a NIC is directly received in the thread of vswitchd. A received data packet is directed to match rules in the accurate flow table of vswitchd and is then sent from the specified port by executing the OpenFlow action and instruction. The advantage of DPDK is that it improves the port I/O performance. In addition, packet sending and receiving and openflow-based data forwarding are implemented in user mode, reducing the overhead caused by switching between kernel mode and user mode and improving the network I/O performance. Compared with SR-IOV, advanced features, such as live migration and NIC hot-add, are supported. The following figure shows the details.

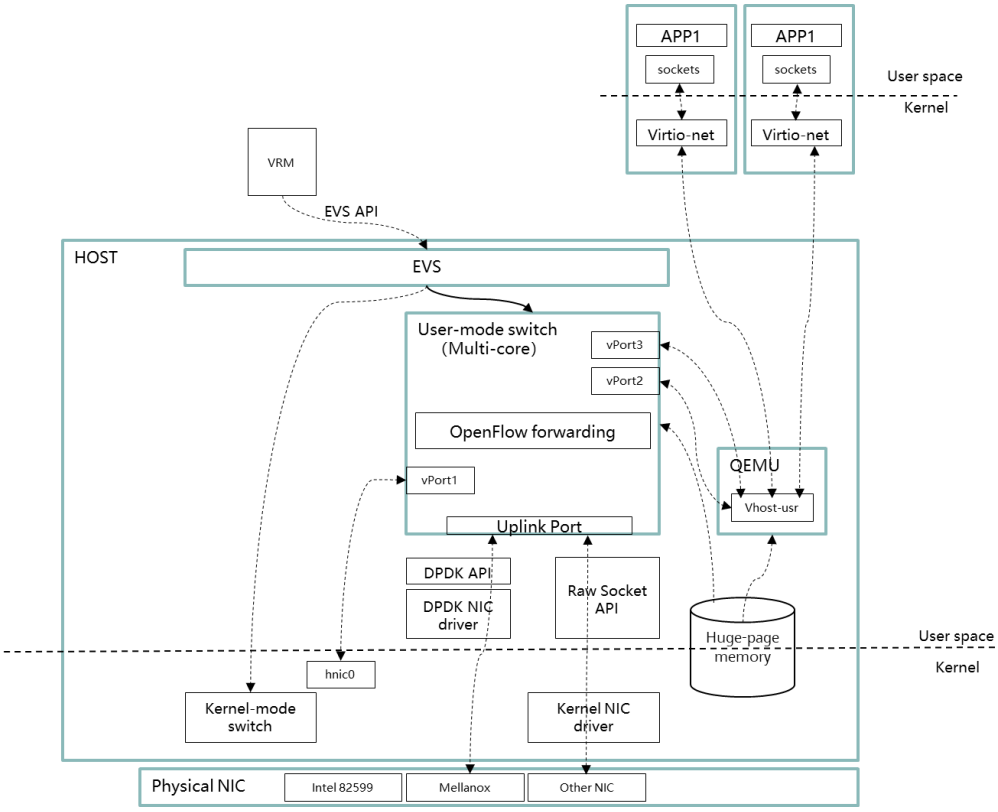


Figure 3-26 User mode

After performing the operations described in section 3.2.4 in the lab guide, clarify the following points:

1. What is the port group used for?
2. How does VRM collect the traffic generated by each port?

4 Storage Basics for Cloud Computing

Hard disks are an indispensable part of personal computers (PCs), both laptops and desktop computers. When we buy a computer, we check out the capacity (for example, 1 TB or 500 GB) and type of its hard disk. Solid-state drive (SSD) and hard disk drive (HDD) are two mainstream hard disk types today. In cloud computing, disks are also an indispensable part. However, unlike the disks in common PCs, the disks in cloud computing are not physical entities. Cloud users concern themselves only with the disk performance and capacity, rather than their physical attributes. However, as a cloud computing engineer, you need to know not only the end users' concerns but also how to convert physical hard disks to cloud disks.

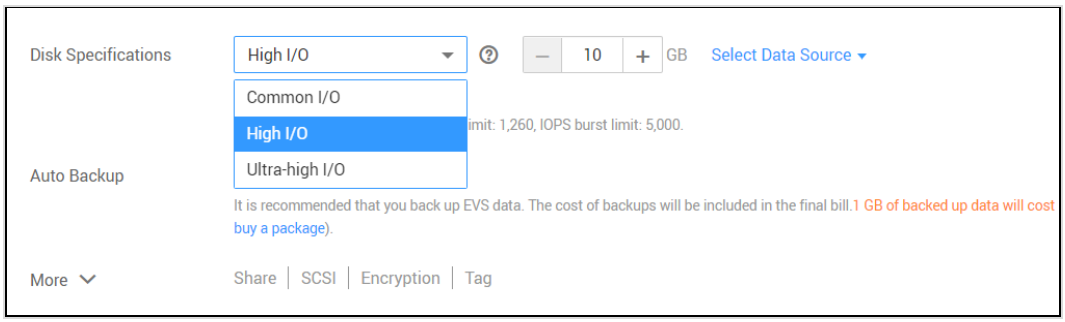


Figure 4-1 Cloud disk specifications

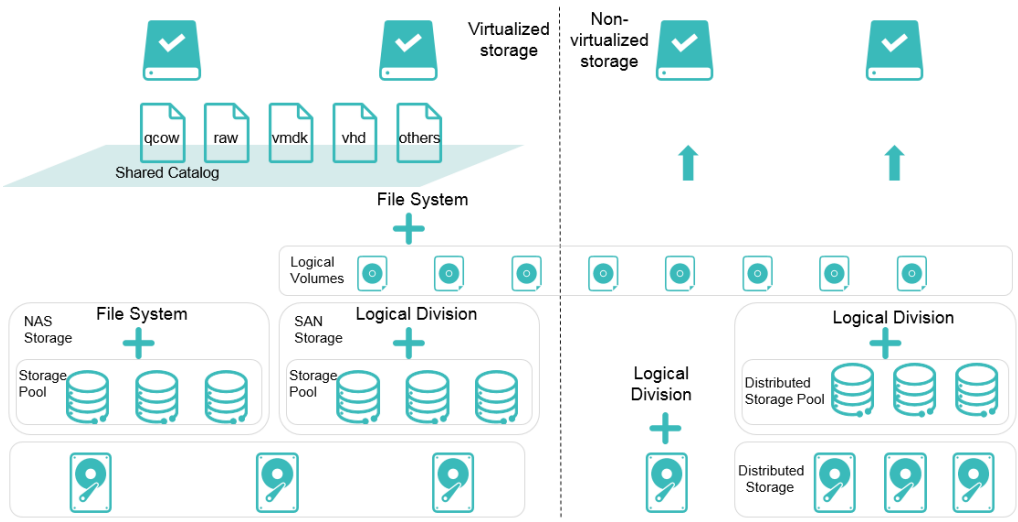


Figure 4-2 Storage architecture in virtualization

In this architecture, physical disks are located at the bottom layer and cloud disks the top layer. A series of operations such as logical partitioning and file system formatting are performed between the two layers. You need to understand these operations. In addition, you need to understand the differences between virtualized and non-virtualized storage. First, let's see the mainstream physical disk types.

4.1 Mainstream Physical Disk Types

4.1.1 HDD

The world's first disk storage system IBM 305 RAMAC was introduced by IBM in 1956. It weighed about a ton and stored five million six-bit characters (5 MB) on a stack of fifty 24-inch disks. In 1973, IBM developed a new type of HDD, IBM 3340. This type of HDD had several coaxial metal platters coated with magnetic material. They were sealed in a box together with removable heads that read changes in magnetic signals from the rotating platters. IBM 3340 was introduced with an advanced disk technology known as "Winchester" and is considered as the predecessor of all of today's HDDs. IBM 3340 consists of two 30 MB storage units, including 30 MB of fixed storage and 30 MB of removable storage. The famous Winchester rifle also had a caliber and charge of 30-30. This type of disk drive was also called "Winchester disk drive." In 1980, Seagate produced the first Winchester disk drive used for PCs, which was approximately the size of a floppy disk drive at the time: 5 MB. Figure 4-3 shows a Winchester disk drive.

Figure 4-3 Winchester disk drive

The read speed of the disk drive was limited by the rotation speed. Increasing the rotation speed could increase the data access speed. However, the platters were paired with magnetic heads and in contact with each other. A high rotation speed may easily damage the disk drive. Therefore, technicians envisioned to make the heads "fly" above the platters. High-speed rotation of the platters produced a flowing wind. Therefore, as long as the heads were properly shaped, they could fly above the platter surfaces like an airplane. In this way, the platters could rotate at a high speed without friction. This was the Winchester technology.

The magnetic heads of a Winchester disk drive are arranged on an actuator arm that moves radially along the platters and do not make contact with the platters. When the heads move in relation to the platters, the heads could induce the magnetic poles on the platter surfaces, and

record or change the status of the magnetic poles to complete data read and write. Because the heads moved at a high speed relative to the platters and they were close to each other, even a particle of dust may damage the disk. Therefore, the disk needs to be encapsulated in a sealed box to maintain a clean internal environment and ensure that the heads and the platters can work efficiently and reliably.

A modern PC typically contains storage media such as an HDD, a CD or DVD-ROM drive, a tape drive, or a solid state disk (SSD). HDDs are considered an irreplaceable and important storage device owing to its large capacity, low price, high read speed, and high reliability.

When we talk about a hard disk, we are usually referring to an HDD. It consists of multiple disk platters, a spindle assembly, a floating head assembly, a head actuator mechanism, a front drive control circuit, and interfaces, as shown in Figure 4-4.

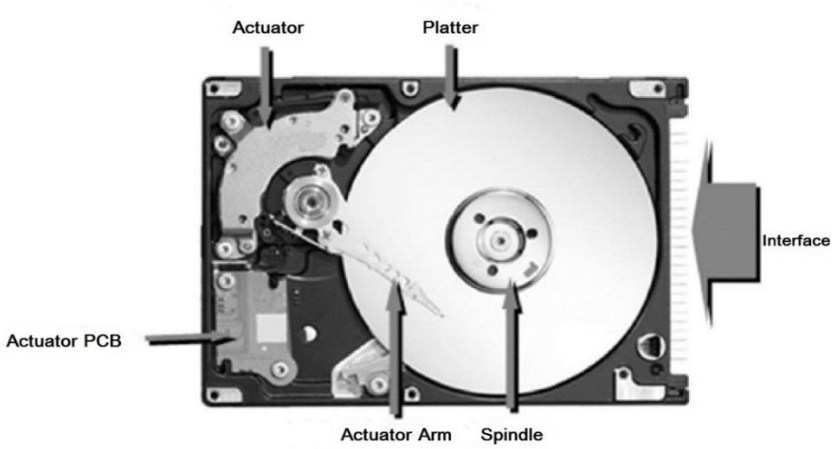


Figure 4-4 HDD assembly

- Platters and spindle. The platters and spindle are two closely connected parts. The platters are circular plates coated with a layer of magnetic material for recording data. The spindle is driven by the spindle motor, which drives the platters to rotate at a high speed.
- Floating head assembly. The floating head assembly consists of read/write heads, an actuator arm, and an actuator axis. When the platters rotate at a high speed, the actuator arm drives the read/write heads at the front end to move in the vertical direction of the rotated platters with the actuator axis as center point. The heads induce the magnetic signals on the platters to read or change the magnetic properties of the magnetic coating and to write information.
- Actuator. It consists of a head actuator, a motor, and a shockproof mechanism. It is used to actuate and precisely locate the heads, so that the heads can read and write data on a specified track at a high speed and with high accuracy.
- Actuator PCB. It is an amplifier circuit sealed in a shielding cavity. It is used to control the induction signals of the heads, adjust the speed of the spindle motor and drive, and locate the heads.
- Interfaces. Typically, they include power supply interfaces and data transfer interfaces. Currently, the mainstream interface types are SATA and SAS.

Platters, metal plates coated with magnetic material, are used to store data in an HDD. Platter surfaces are divided as circles of tracks. When the platters rotate rapidly driven by the motor, the heads arranged on the platter surfaces are precisely controlled, and data is read and written along the tracks. When the system writes data to an HDD, a current that varies with the data content is generated in the heads. The current generates a magnetic field, which changes the

status of the magnetic material on the platter surfaces. The status can be maintained permanently after the magnetic field disappears, which means the data is saved. When the system reads data from an HDD, the heads pass through the specified area of the platters. The magnetic field on the platter surfaces causes the heads to generate an induced current or change the coil impedance. The change is captured and processed to restore the original written data.

Currently, SATA and SAS hard drives are commonly used and distinguished by interface type. The following describes the two types of hard drives.

- SATA hard drive

  You need to know the advanced technology attachment (ATA) interface first. ATA interfaces are integrated drive electronics (IDE) interfaces. ATA interfaces have been developed since the 1980s. Due to low price and good compatibility, ATA interfaces have become the mainstream storage interfaces in the market. However, with the rapid development of technologies, its low speed hinders its application in modern computer systems.

  SATA, that is, serial ATA, has basically replaced all parallel ATA interfaces, as shown in Figure 4-5. As its name implies, SATA transfers data in serial mode. One of its distinctive features is that SATA is faster than ATA. Currently, the mainstream SATA 3.0 can have a transfer rate of 6.0 Gbit/s, which is several times higher than that of parallel ATA.

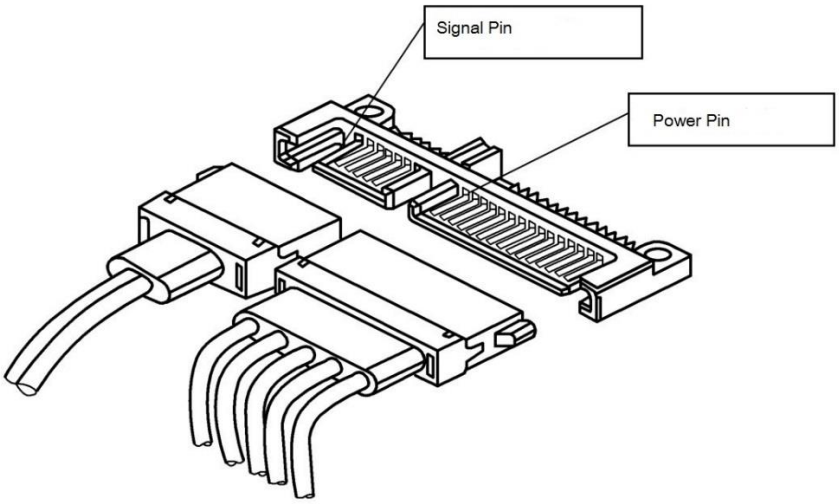


Figure 4-5 SATA interface

SATA uses independent data interfaces and signal interfaces to transfer data. Parallel ATA (PATA) uses a 16-bit data bus for data transfer and needs to transfer many additional support and control signals. Due to the limitation of the manufacturing process, PATA is susceptible to noise and its signal voltage is 5 V. In contrast, SATA uses embedded clock signals and has a stronger error correction capability. Signal voltage of SATA is 0.5 V.

From the perspective of bus structure, SATA uses a single channel to perform point-to-point transfer. Data is transferred by bit in serial mode, and checksum signal bit is embedded in the data. This transfer method ensures the transfer rate and improves the transfer reliability.

SATA uses a point-to-point architecture and supports hot swap. The SATA interface uses seven data pins and fifteen power pins. Compared with the parallel ATA interface, the SATA interface uses thinner cables, which are easy to bend. An SATA cable can have a

maximum length of one meter, facilitating chassis cooling. Figure 4-6 shows the SATA hard drive.

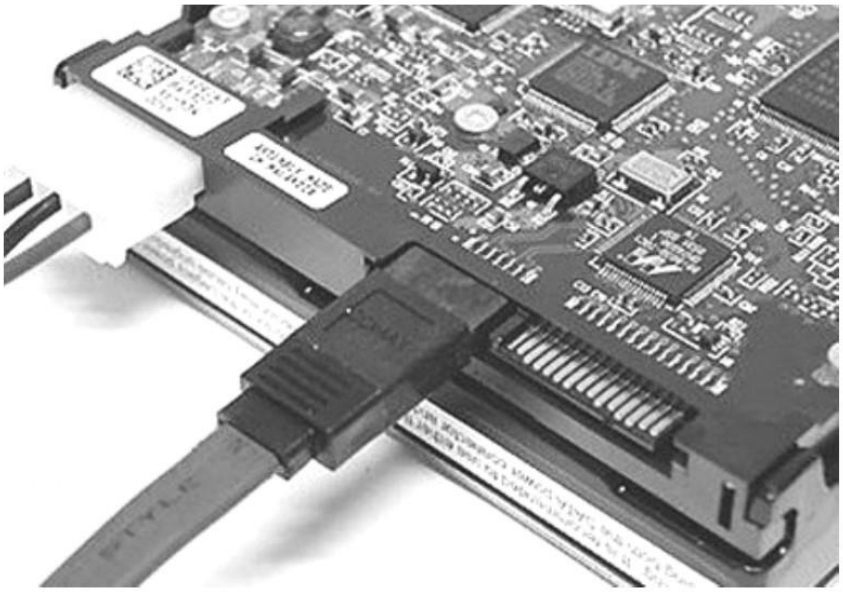


Figure 4-6 SATA interface

- SAS hard drive

  Serial attached SCSI (SAS), that is, small computer system interface. Similar to SATA, SAS is developed based the parallel SCSI technology.

  SCSI is widely used as enterprise storage owing to its high performance. The SCSI interface provides three types of connectors: 50-pin, 68-pin, and 80-pin. After decades of development, the mainstream SCSI technology Ultra 320 SCSI supports a transfer rate of 320 MB/s.

  SAS is a branch of the SCSI technology. Similar to SATA, SAS uses serial transfer to achieve higher performance. Currently, the mainstream SAS transfer rate is 6 Gbit/s. In addition, with serial technology, thin and long cables can be used to achieve a longer connection distance and improve an anti-interference capability. Figure 4-7 shows the front and rear views of a SAS interface.

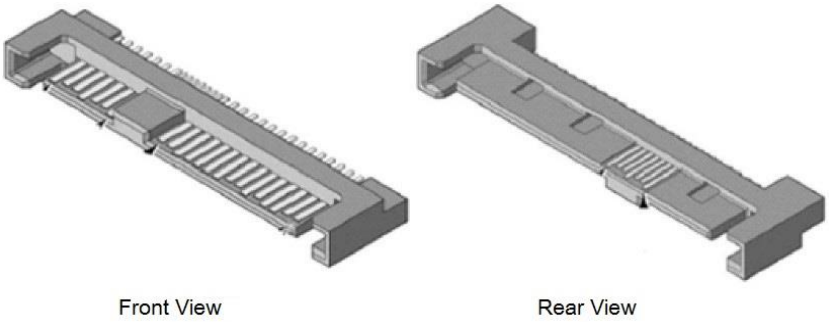


Figure 4-7 SAS interface

SAS is backward compatible with SATA. SAS controllers can be connected to SATA hard drives, which deliver low cost and excellent flexibility to enterprises.

SAS uses a point-to-point architecture. Similar to SATA, SAS does not require termination signals and does not have synchronization issues. SAS supports up to 65,536 devices, whereas SCSI supports only 8 or 16 devices.

- NL-SAS hard drive

  Near Line SAS (NL-SAS) is a type of hard drive between SATA and SAS. Compared with a SATA hard drive, a SAS hard drive has a higher read/write speed, higher price, and smaller capacity. Therefore, an NL-SAS hard drive is developed, which consists of SAS interfaces and SATA platters, as shown in Figure 4-8. An HDD consists of SAS interfaces and SATA platters.

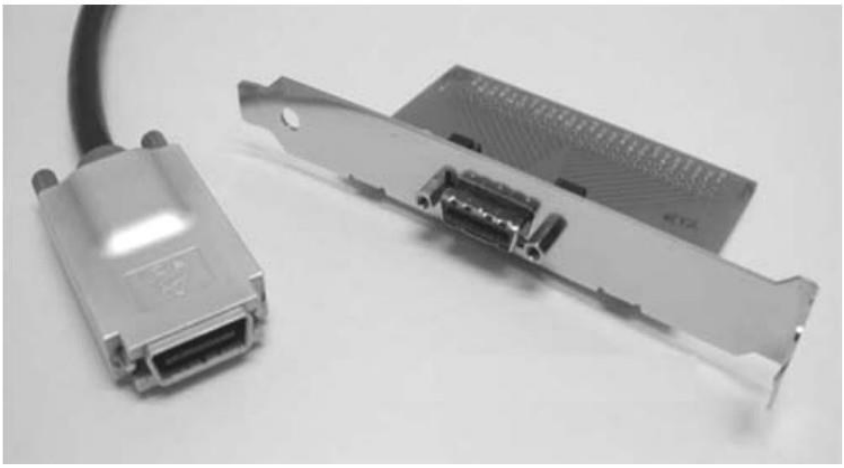


Figure 4-8 NL-SAS

The input/output operations per second (IOPS) of an NL-SAS hard drive is approximately half of that of a SAS hard drive that has an IOPS of 15000. However, the NL-SAS hard drive outperforms the SATA hard drive. In addition, the NL-SAS hard drive has the capacity and price of the SATA hard drive and the reliability of the SAS hard drive. Therefore, the NL-SAS hard drive is popular in the market.

4.1.2 SSD

The world's first SSD appeared in 1989. SSDs were extremely expensive at that time, but its performance was far lower than that of a common HDD. Therefore, it was not widely used. However, due to its unique features such as shock resistance, low noise, and low power consumption, it has been used extensively in some special fields such as medical and military.

With the maturing of technology, improvement of manufacturing process, and cost reduction, SSD gained increasing popularity in the consumer field. In 2006, Samsung launched its first laptop with a 32 GB SSD. At the beginning of 2007, SanDisk launched two 32 GB SSD products. In 2011, a severe flood occurred in Thailand. Many HDD manufacturers, such as Western Digital and Seagate, were forced to close their factories in Thailand. As a result, the shipments of HDDs dropped sharply and prices increased sharply. This greatly stimulated the demand for SSDs, bringing the golden age of SSDs. Today, the capacity, cost, speed, and service life of SSDs have been greatly improved compared with the original products. The capacity of common SSDs in the market has reached 128 GB and 256 GB. The price per GB is only a fraction of the price at that time, making SSDs affordable to common consumers. SSDs are one of the most essential storage devices in the ultra-thin laptop and tablet fields. It is foreseeable that SSDs will receive even greater attention in the next few years.

An SSD consists of controller and memory chips. Simply put, an SSD is a hard drive composed of a solid state electronic chip array. The interface specifications, definitions, functions, and usage of SSDs are the same as those of common HDDs. The appearance and dimensions of SSDs are the same as those of common HDDs, including 3.5-inch, 2.5-inch, and 1.8-inch. Because an SSD does not have a rotation structure like a common HDD, it has

superior shock resistance and wide operating temperature range (–45°C to +85°C). Therefore, it is widely used in fields like the military, vehicle-mounted, industrial control, video surveillance, network surveillance, network terminal, electric power, medical care, aviation, and navigation equipment. Traditional HDDs are disk drives and data is stored in disk sectors. The common storage medium of an SSD is flash memory. SSDs are one of the major trends of hard drives in the future. Figure 4-9 shows the internal structure of an SSD.

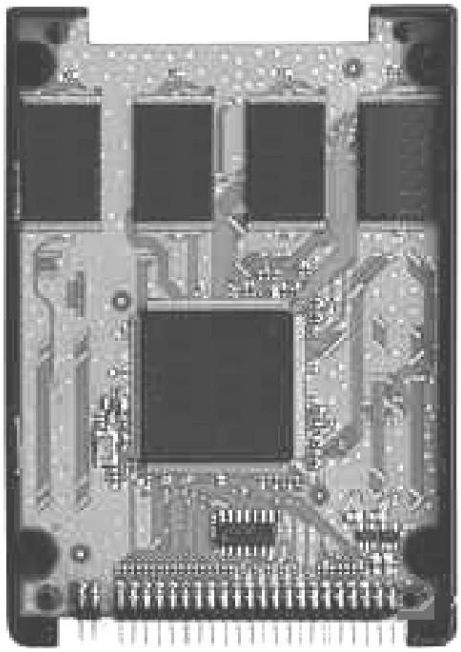


Figure 4-9 Internal structure of an SSD

An SSD consists of a flash controller and memory chips. The flash controller controls the coordination of the data read/write process and the memory chips are responsible for storing data. Memory chips are classified as two types by medium. A most common type is to use a flash memory chip as the storage medium, and the other type is to use a dynamic random access memory (DRAM) chip as the storage medium.

- Flash-based SSDs

  The most common SSDs use a flash memory chip as the storage medium. Flash memory chips can be manufactured into various electronic products, such as SSDs, memory cards, and USB drives. These devices are small in size and easy to use. The SSDs discussed in this section are flash-based SSDs.

- DRAM-based SSDs

  This type of SSDs uses DRAM as the storage medium. This type of storage medium has superior performance and long service life and is currently widely used in memory. However, the DRAM stores data only when it is powered on. Once it is powered off, information stored in the DRAM will be lost. Therefore, the DRAM requires extra power supply for protection. Currently, such SSDs are expensive and used in a few fields.

SSDs have the following advantages over traditional HDDs:

- High read speed

  As an SSD uses flash memory chips as the storage medium and has no disk and motor structure, the seek time is saved when data is read, and the speed advantage can be particularly reflected when data is read randomly. In addition, the SSD performance is not affected by disk fragments.

- Superior shock resistance

  There are no moving mechanical parts inside an SSD, which eliminates the possibility of a mechanical fault and enables SSDs to tolerate collisions, shocks, and vibrations. SSDs run properly even when they are moving fast or tilted severely, and minimize data loss when its laptop drops or collides into another object.
- No noise

  There is no mechanical motor inside an SSD, which means it is truly noise-free and silent.
- Small size and lightweight

  An SSD can be integrated on a small printed circuit board (PCB).
- Wider operating temperature range

  A typical HDD can work only within an operating temperature range of 5°C to 55°C. Most SSDs can work within an operating temperature range of –10°C to 70°C, and some industrial-grade SSDs can work within an operating temperature range of –40°C to 85°C or even larger.

However, SSDs also have two disadvantages, and therefore cannot be used as substitutes for HDDs. One disadvantage is the high cost. Currently, the price per GB of an SSD is about 10 times that of a traditional HDD. Large-capacity SSDs are still rare in the market. Therefore, for applications that are insensitive to the data read/write speeds, HDDs are still the first choice. The other disadvantage is limited service life. Typically, a high performance flash memory can be erased for 10,000 to 100,000 times, and a common consumer-grade flash memory can be erased for 3,000 to 30,000 times. With continuous improvement of the manufacturing process, a smaller size of the storage unit further reduces the maximum erase times of flash memory. Typically, the controller of the SSD is able to balance chip loss, so that the storage chip can be consumed more evenly, thereby improving the service life.

SSDs, as storage media with higher read/write speeds than traditional HDDs, have received widespread attention. Unlike traditional HDDs, SSDs do not have any mechanical components. Therefore, SSDs improve performance quickly. In addition, SSDs have distinctive features such as shock resistance, small size, no noise, and low cooling requirements. Many people hope that SSDs can replace traditional HDDs and become a new generation of storage devices. However, the cost of SSDs is far higher than that of traditional HDDs and the performance of HDDs can meet a large part of requirements. Therefore, the traditional HDDs and the SSDs will coexist and develop together for a long time.

Table 4-1 Comparison of different types of hard drives

Item	SSD	SAS	NL-SAS	SATA
Performance	Very high	High	Relatively high	Relatively high
Reliability	Minor	High	Relatively high	Minor
Price	Very high	High	Relatively low	Low
Power consumption	Minor	High	Relatively low	Relatively low

Item	SSD	SAS	NL-SAS	SATA
Recommended scenario	Suitable for users with very large data access	Suitable for high- and mid-end users with discrete data	Suitable for users with large data blocks and low service I/O pressure	Suitable for users with large data blocks and low service pressure

4.2 Centralized Storage and Distributed Storage

4.2.1 Centralized Storage

Centralized storage means that all storage resources are centrally deployed and are provisioned over a unified interface. With centralized storage, all physical disks are centrally deployed in disk enclosures and are used to provide storage services externally through the controller. Centralized storage typically refers to disk arrays.

Based on the technical architecture, centralized storage can be categorized as SAN and NAS. SAN can be further categorized as FC SAN, IP SAN, and FCoE SAN. Currently, FC SAN and IP SAN technologies are mature, and FCoE SAN still has a long way to go to reach maturity.

A disk array combines multiple physical disks into a single logical unit. Each disk array consists of one controller enclosure and multiple disk enclosures. This architecture delivers an intelligent storage space featuring high availability, high performance, and large capacity.

SAN Storage

The storage area network (SAN), with blocks as basic data access unit, is a dedicated high-speed storage network that is independent of the service network system. This type of network is implemented in the form of fibre channel storage area network (FC SAN), IP storage area network (IP SAN), and serial attached SCSI storage area network (SAS SAN). Different implementations transfer data, commands, and status between servers and storage devices using different communication protocols and connections.

Direct attached storage (DAS) is the most widely used storage system before SAN is introduced. DAS has been used for nearly forty years. Early data centers used disk arrays to expand storage capabilities in DAS mode. Storage devices of each server serve only a single application and provide an isolated storage environment. However, these isolated storage devices are difficult to share and manage. With the increase of user data, the disadvantages of this expansion mode in terms of expansion and disaster recovery are increasingly obvious. SAN resolves these issues. SAN connects these isolated storage islands through a high-speed network. These storage devices can be shared by multiple servers through the network, delivering remote data backup and excellent scalability. All these factors make this storage technology develop rapidly.

As an emerging storage solution, SAN accelerates data transfer, delivers greater flexibility, and reduces network complexity, alleviating the impact of transfer bottlenecks on the system and improving the efficiency of remote disaster recovery.

A SAN is a network architecture that consists of storage devices and system components, including servers that need to use storage resources, host bus adapters (HBAs) that connect storage devices, and FC switches.

On a SAN, all communication related to data storage is implemented on an independent network that is isolated from the application network, which means that transferring data on the SAN does not affect the data network of the existing application system. Therefore, SAN delivers higher I/O capabilities of the entire network without reducing the data network efficiency of the original application system, redundant links to the storage system, and support for the high availability (HA) cluster system.

With the development of SAN technologies, three SAN types are made available: FC SAN, IP SAN, and SAS SAN. The following describes FC SAN and IP SAN.

In FC SAN, two network interface adapters are configured on the storage server. One is a common network interface adapter (NIC) that connects to the service IP network and the server interacts with the client through this NIC. The other is an HBA that connects to FC SAN and the server communicates with the storage devices on the FC SAN through this adapter. Figure 4-10 shows the FC SAN architecture.

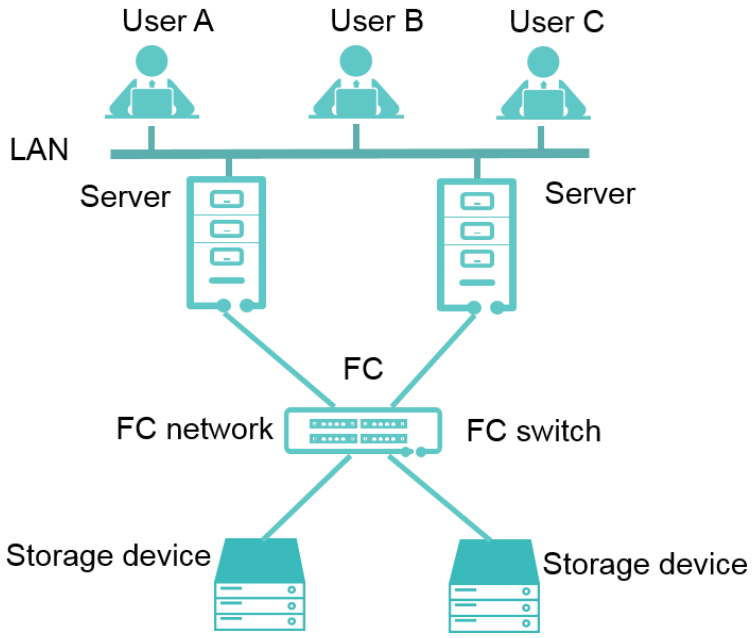


Figure 4-10 FC SAN architecture

IP SAN has become a popular network storage technology in recent years. The early SANs are all FC SANs, where data is transferred in the fibre channel as a block-based access unit. Due to the incompatibility between FC protocol and the IP protocol, customers who want to implement the FC SAN have to purchase its devices and components. Its high price and complicated configuration impede large number of small and medium-sized users' demands for it. Therefore, FC SAN is mainly used for mid and high-end storage that requires high performance, redundancy, and availability. To popularize SANs and leverage the advantages of SAN architecture, technicians consider to combine SANs with prevailing and affordable IP networks. Therefore, the IP SAN that uses the existing IP network architecture is introduced. The IP SAN is a combination of the standard TCP/IP protocol with the SCSI instruction set and implements block-level data storage based on the IP network.

The difference between IP SAN and FC SAN lies in the transfer protocol and medium. Common IP SAN protocols include iSCSI, FCIP, and iFCP. iSCSI is the fastest growing protocol standard. In most cases, IP SAN refers to iSCSI-based SAN.

An iSCSI initiator (server) and an iSCSI target (storage device) form a SAN. Figure 4-11 shows the IP SAN architecture.

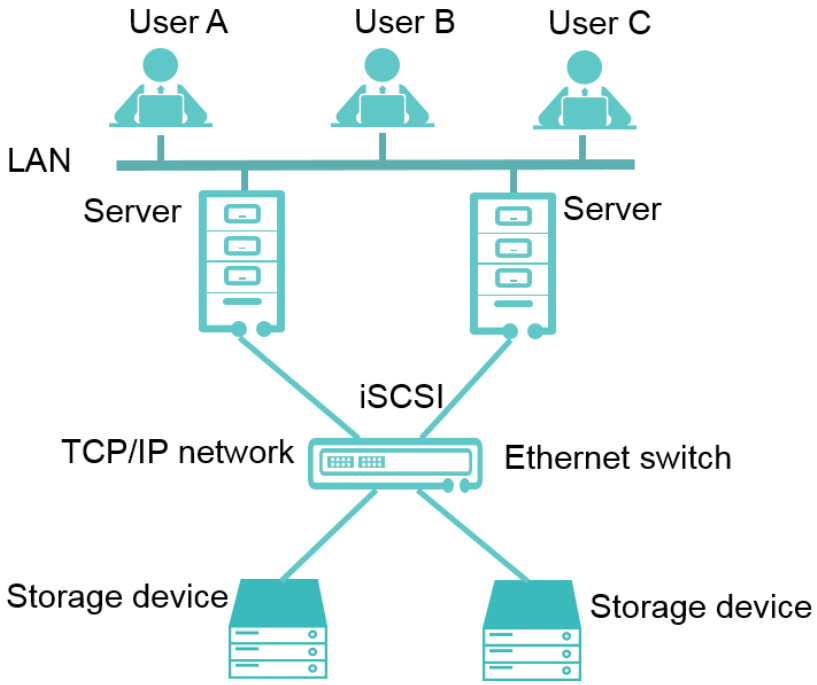


Figure 4-11 IP SAN architecture

IP SAN has the following advantages over FC SAN:

- Standard access. IP SANs require only common Ethernet cards and switches for connection between storage devices and servers instead of dedicated HBAs or fibre channel switches.
- Long transfer distance. IP SANs are available wherever IP networks exist. Currently, IP networks are the most widely used networks in the world. Good maintainability. Most network maintenance personnel have good knowledge of IP networks, therefore, IP SANs are more acceptable than FC SANs. In addition, IP SAN can be maintained using the developed IP network maintenance tools.
- The bandwidth can be easily expanded. With the rapid development of 10 GB Ethernet, it is inevitable that the bandwidth of a single port on the Ethernet-based iSCSI IP SAN can be expanded to 10 GB.

These advantages reduce the total cost of ownership (TCO). For example, to build a storage system, the TCO includes purchase of disk arrays and access devices (HBAs and switches), personnel training, routine maintenance, capacity expansion, and disaster recovery capacity expansion. With the wide application of IP networks, IP SANs help customers significantly cut down the purchase cost of access devices, maintenance cost, and capacity and network expansion costs.

Table 4-2 lists the comparison between IP SAN and FC SAN.

Table 4-2 Comparison between IP SAN and FC SAN

Item	IP SAN	FC SAN
Network speed	1 GB, 10 GB, and 40 GB	4 GB, 8 GB, and 16 GB
Network architecture	Existing IP networks	Dedicated fibre channel networks and HBAs
Transfer distance	Not limited theoretically	Limited by the maximum transfer distance of optical fibers
Management and maintenance	As simple as operating IP devices	Complicated technologies and management
Compatibility	Compatible with all IP network devices	Poor compatibility
Performance	1 GB and 10 GB being developed	Very high transfer and read/write performance
Cost	Low purchase and maintenance costs	High purchase (Fibre Channel switches, HBAs, Fibre Channel disk arrays, and so on) and maintenance (personnel training, system configuration and supervision, and so on) costs
Disaster recovery	Local and remote DR available based on existing networks at a low cost	High hardware and software costs for disaster recovery (DR)
Security	Relatively low	Relatively high

NAS

Network attached storage (NAS) is a technology that integrates distributed and independent data into a large-scale and centralized data center for access by different hosts and application servers. NAS is a file-level computer data storage connected to a computer network providing data access to heterogeneous group of clients. An NAS server contains storage devices, such as disk arrays, CD/DVD drives, tape drives, or portable storage media. An NAS server delivers an embedded OS to share files across platforms.

The inception of NAS is closely related to the development of network. After the inception of the ARPANET, modern network technologies develop rapidly, and users have increasing demand for sharing data over network. However, sharing files on the network is confronted with many issues such as cross-platform access and data security. Figure 4-12 shows network sharing in the early stage.

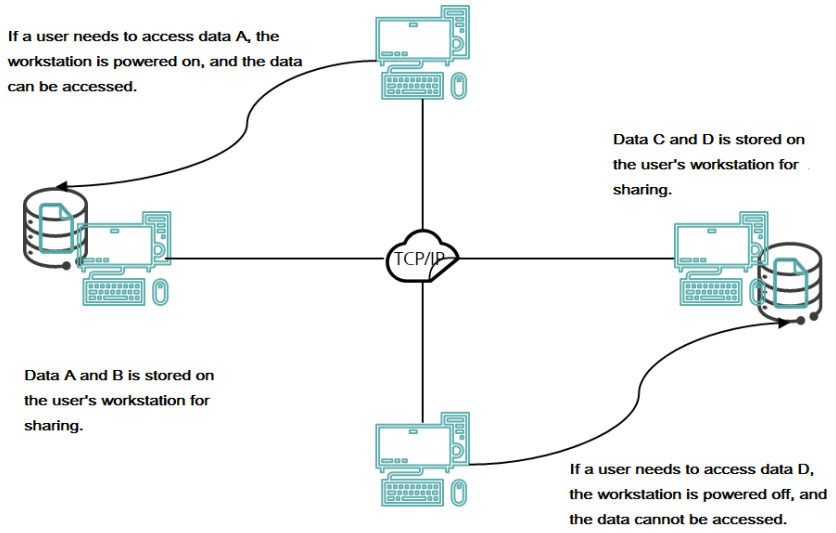


Figure 4-12 Network sharing in the early stage

To resolve this issue, technicians use a dedicated computer to store a large number of shared files. The computer is connected to an existing network and allows all users over the network to share the storage space. In this way, the early UNIX network environment evolved into a way of sharing data depending on file servers.

Storing shared data using dedicated servers with large storage space must ensure data security and reliability. A single server needs to process access requests from multiple servers. Therefore, I/O performance of the file server needs to be optimized. In addition, the extra overhead of the OS is unnecessary. Therefore, the server used in this mode should run a thin OS with only the I/O function and be connected to an existing network. Users over the network can access the files on this special server as they access the files on their own workstations, meeting the demands for sharing files over the network of all users. Figure 4-13 shows the TCP/IP network in the early UNIX environment.

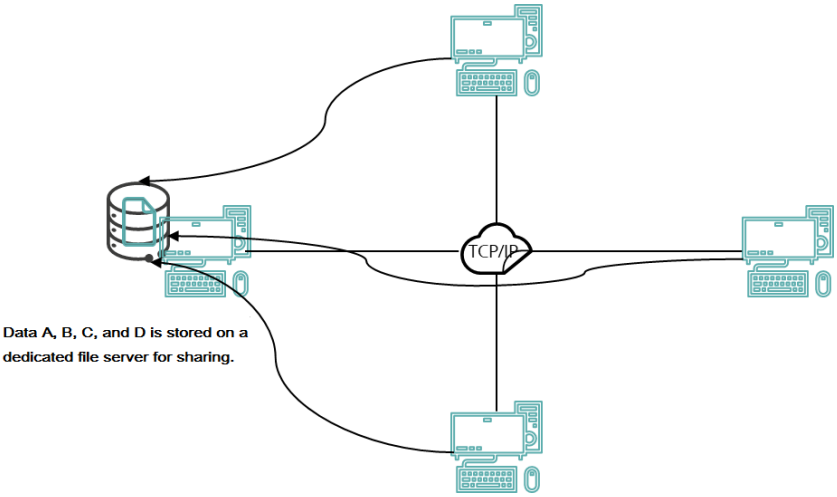


Figure 4-13 TCP/IP network sharing

With the development of networks, there are increasing demands for sharing data among computers over the network. People want system and users in a network to be connected to a specific file system to obtain access to remote files from a shared computer as they access files in a local OS. This way, they can use a virtual file set with files stored in a virtual

location instead of a local computer. This storage mode develops towards integrating with traditional client or server environment that supports Windows OS. This involves Windows network capabilities, proprietary protocols, and UNIX-based database servers. In the initial development phase, the Windows network consists of a network file server that is still in use and uses a dedicated network system protocol. The following shows the early Windows file server.

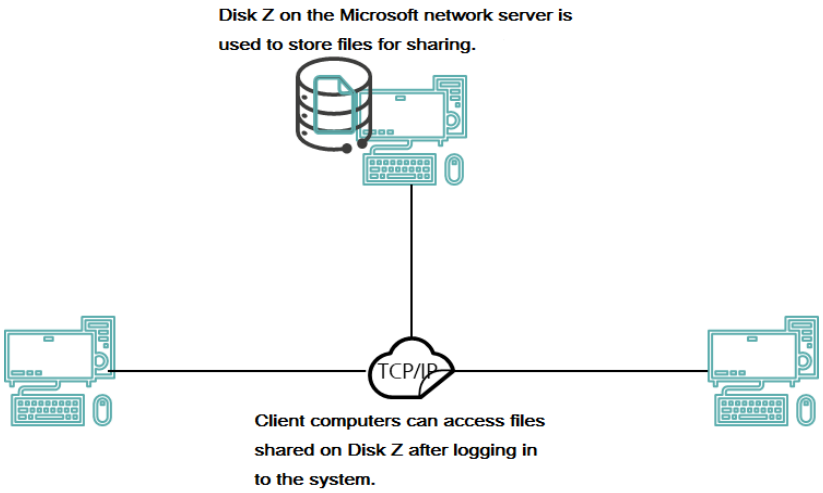


Figure 4-14 Windows file server

The inception of file servers drives centralized data storage, leading to sharp growth of centralized data and service volume. Therefore, NAS products dedicated to file sharing services have been developed.

NAS typically has its own nodes on a LAN and does not require the intervention of application servers. NAS allows users to directly access file data over the network. In this configuration, NAS centrally manages and processes all shared files on the network and releases loads from application or enterprise servers, reducing the TCO and maximizing customers' ROI. Simply speaking, a NAS is a device that is connected to the network with file storage function. Therefore, NAS is also called a network file storage device. It is a dedicated file data storage server. It delivers centralized file storage and management and separates storage devices from servers, releasing bandwidth, improving performance, maximizing customers' ROI, and reducing the TCO.

Essentially, NAS is a storage device rather than a server. NAS is not a compact file server. It delivers more distinctive features than other servers. Servers process services and storage devices store data. In a complete application environment, the two types of devices must be combined.

The advantage of NAS is that it can deliver file storage services in a fast and cost-effective manner using existing resources in the data center. The current solution is compatible between UNIX, Linux, and Windows OSs, and can be easily connected to users' TCP/IP networks. The following shows the NAS system.

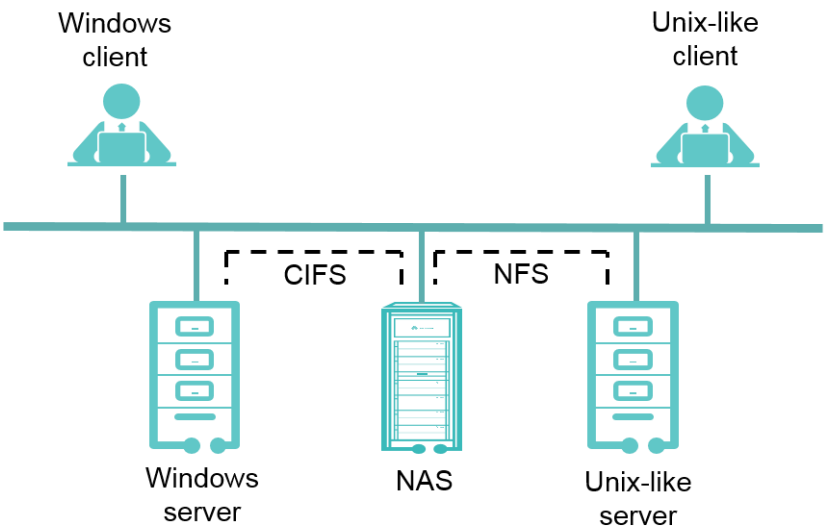


Figure 4-15 NAS system

NAS should be able to store and back up large volumes of data, and deliver stable and efficient data transfer services. These requirements cannot be fulfilled only by hardware. Therefore, NAS is software-dependent. The NAS software can be divided as five modules: OS, volume manager, file system, file sharing over a network, and web management, as shown in Figure 4-16.

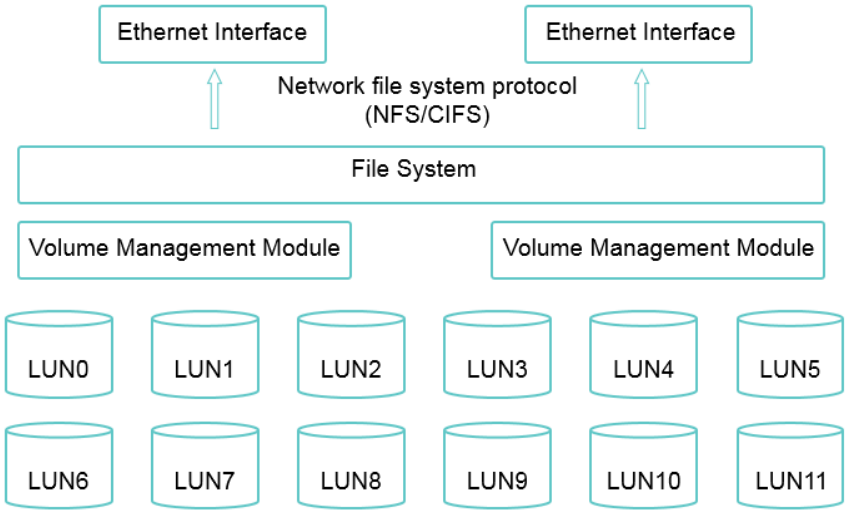


Figure 4-16 NAS architecture

The NAS device can read and write the common internet file system (CIFS) or the network file system (NFS), and can also read and write the two systems at the same time.

CIFS is a public and open file system developed by Microsoft Server Message Block (SMB). SMB is a file sharing protocol set by Microsoft based on NetBIOS. Users can access data on a remote computer through CIFS. In addition, CIFS prevents read-write conflicts and write-write conflicts to support multi-user access.

To enable Windows and Unix computers to share resources and enable Windows users to use resources on Unix computers as if they use Windows NT servers, the best way is to install software that supports the SMB/CIFS protocol on Unix computers. When all mainstream OSs

support CIFS, communications between computers are convenient. Samba helps Windows and Unix users achieve this desire. A CIFS server is set up to share resources with the target computers. The target computers mount the shared resources on the CIFS server to their own OSs through simple shared mapping and use the resources as local file system resources. Through a simple mapping, the computer users obtain all the required shared resources from the CIFS server.

Network File System (NFS) is a distributed file system protocol originally developed by Sun Microsystems in 1984, allowing a user on a client computer to access files over a computer network much like local storage is accessed. It is designed for use among different operating systems, therefore, its communication protocol is independent from hosts and operating systems. When users want to use remote files, they only need to use the mount command to mount the remote file system under their own local file systems. There is no difference between using remote files and local files.

The platform-independent file sharing mechanism of NFS is implemented based on the XDR/RPC protocol.

External data representation (EDR) can convert the data format. Typically, EDR converts data into a unified standard data format to ensure data consistency between different platforms, operating systems, and programming languages.

The remote procedure call (RPC) requests services from remote computers. Users send requests to remote computers over the network. The remote computers process the requests.

NFS uses the virtual file system (VFS) mechanism to send users' remote data access requests to servers through unified file access protocols and remote procedure calls. NFS has evolved. Since its inception, it has been updated in four versions and has been ported to almost all mainstream operating systems, becoming the de facto standard for distributed file systems. NFS is introduced in an era when the network status is unstable. It is initially transmitted based on UDP rather than TCP with higher reliability. UDP works well on a LAN with better reliability, however, it is incompetent on a WAN with poor reliability, such as the Internet. Currently, TCP enhancements make NFS using TCP deliver high reliability and good performance.

Table 4-3 lists the comparison between CIFS and NFS.

Table 4-3 Comparison between CIFS and NFS

Item	CIFS	NFS
Transfer characteristics	Network-based, requiring high reliability	Independent transfer
Ease of use	Requiring no additional software	Requiring dedicated software
Security	Fault recovery unavailable	Fault recovery available
File conversion	File format not reserved	File format reserved

4.2.2 RAID

In a centralized storage system, all disks are put into disk enclosures and uniformly managed by the controller enclosure. The system supports dynamic storage capacity expansion and improves fault tolerance as well as read and write performance. Such a system typically uses a technology called Redundant Arrays of Independent Disks (RAID).

There are seven basic RAID levels: RAID 0 to RAID 6. There are also common combinations of basic RAID levels, such as RAID 10 (combination of RAID 1 with RAID 0) and RAID 50 (combination of RAID 5 with RAID 0). Different RAID levels represent different storage performance, data security, and costs. This section describes only RAID 0, RAID 1, RAID 5, and RAID 6.

- RAID 0

  RAID 0, also known as stripping, combines multiple physical disks into a logical disk, which delivers the highest storage performance among all RAID levels. Although RAID 0 delivers the highest speed, it has no redundancy and does not support parallel I/O. When data is stored, data is segmented based on the number of disks that build the RAID 0 volume, and the data is written into the disks in parallel. Therefore, RAID 0 is the fastest among all levels. However, RAID 0 has no redundancy. If a physical disk fails, all data will be lost.

  Theoretically, the total disk performance equals the performance of a single disk multiplied by the number of disks. However, due to the bus I/O bottleneck and other factors, the RAID performance is not a multiple of the number of disks. That is, if the performance of one disk is 50 MB/s, the RAID 0 performance of two disks is about 96 MB/s, and that of three disks may be 130 MB/s instead of 150 MB/s, therefore, the RAID 0 performance of two disks is significantly improved.

  Figure 4-17 shows RAID 0. There are two disks, Disk 1 and Disk 2. RAID 0 divides the data (D1, D2...) into two parts and stores the two parts at the same time. D1 and D2 are stored in Disk 1 and Disk 2, respectively. After D1 is stored, D3 is stored in Disk 1. Other data blocks are stored in the same way. In this way, two disks can be considered as a large disk, and I/O is simultaneously performed on both disks. However, if a data block is damaged, the entire data will be lost.

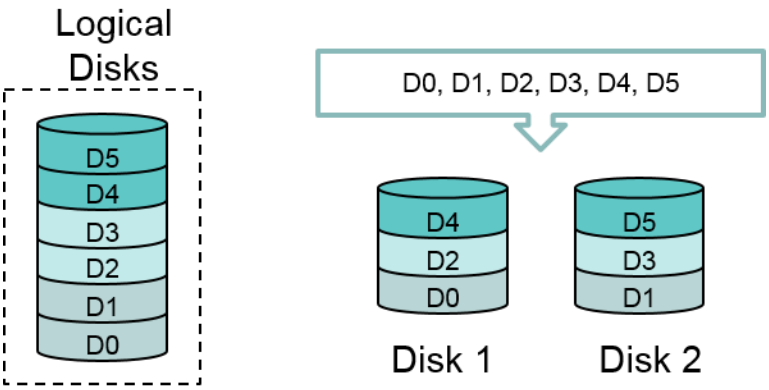


Figure 4-17 RAID 0

  RAID 0 delivers superior read and write performance but has no data redundancy. It is applicable to applications that have fault tolerance for data access and applications that can regenerate data through other methods, such as web applications and streaming media.

- RAID 1

RAID 1, also known as mirror or mirroring, is designed to maximize the availability and repairability of user data. RAID 1 automatically copies all data written to one disk to the other disk in a RAID group.

RAID 1 writes the same data to the mirror disk while storing the data on the source disk. When the source disk fails, the mirror disk takes over services from the source disk. RAID 1 delivers the best data security among all RAID levels because the mirror disk is used for data backup. However, no matter how many disks are used, the available storage space is only the capacity of a single disk. Therefore, RAID 1 delivers the lowest disk usage among all RAID levels.

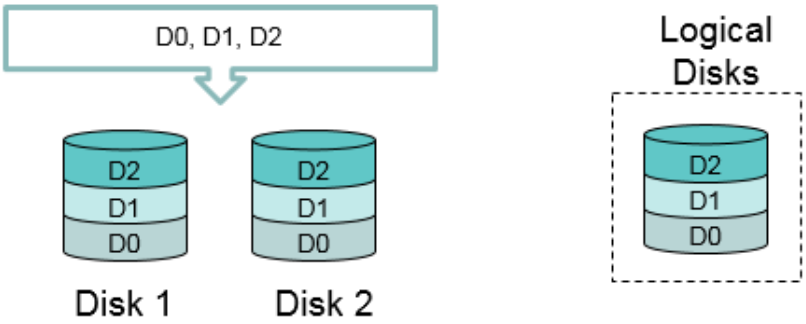


Figure 4-18 RAID 1

Figure 4-18 shows RAID 1. There are two disks, Disk 1 and Disk 2. RAID 1 stores the data (D1, D2...) in the source disk (Disk 1), and then stores the data again in Disk 2 for data backup.

RAID 1 is the most expensive storage unit among all RAID levels. However, it delivers the highest data security and availability. RAID 1 is applicable to online transaction processing (OLTP) applications with intensive read operations and other applications that require high read/write performance and reliability, for example, email, operating system, application file, and random access environment.

- RAID 5

  RAID 5 is the most common RAID level in advanced RAID systems and is widely used for its superior performance and data redundancy balance design. It is short for independent data disks with distributed parity. RIAD 5 uses parity for parity check and error correction.

  Figure 4-19 shows the data storage mode of RAID 5. In the figure, three disks are used as an example. **P** is the check value of data, and **D** is the actual data. RAID 5 does not back up the stored data but stores corresponding parity data on different member disks. When the data on a member disk is corrupted, the corrupted data can be recovered based on the data on other member disks. Therefore, RAID 5 is a storage solution that balances storage performance, data security, and storage costs.

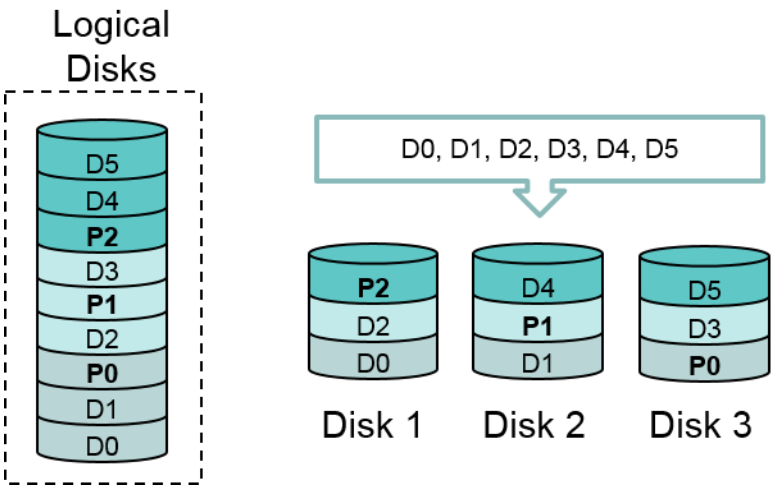


Figure 4-19 RAID 5

RAID-5 is a widely used data protection solution that delivers optimal overall performance although it has some capacity loss. It applies to I/O-intensive applications with a high read/write ratio, such as OLTP applications.

- RAID 6

  RAID 6 is a RAID mode designed to further enhance data protection. Compared with RAID 5, RAID 6 has an independent parity block. In this way, each data block has two parity blocks (one is hierarchical check and the other is overall check). Therefore, RAID 6 delivers superior data redundancy performance. However, two parity check mechanisms slow down data writes, the RAID controller design is more complicated, and two parity areas reduce available storage space.

  Common RAID 6 technologies include PQ and DP. The two technologies use different methods for obtaining verification information, but both technologies allow data loss on two disks in an array.

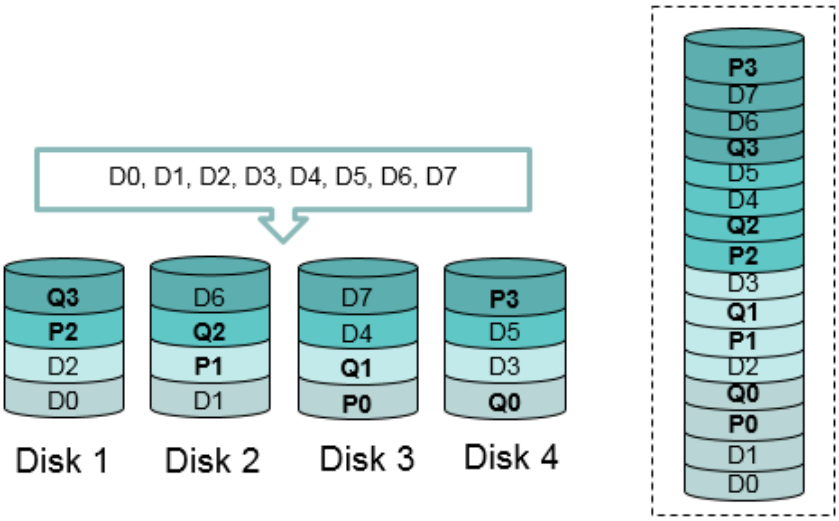


Figure 4-20 RAID 6

Data security of RAID 6 is higher than that of RAID 5. Even if two disks in an array fail, the array can still work and recover the data on the faulty disks. However, the design of the controller is more complicated, the write speed is lower, and it takes longer time to calculate

check information and verify data correctness. When write operations are performed on each data block, two independent check calculations need to be performed, resulting heavier system load. In addition, the disk usage is lower, and the configuration is more complicated. Therefore, RAID 6 is applicable to the environment that requires high data accuracy and integrity.

NOTE

Section 4.1 and 4.2 are from *Information Storage and IT Management* of Huawei Certification books.

4.2.3 Distributed Storage and Replication

Distributed storage is quite different from the conventional storage. It virtualizes all available space distributed across different hosts into a single virtual device. The data stored in this virtual storage is also distributed all over the storage network.

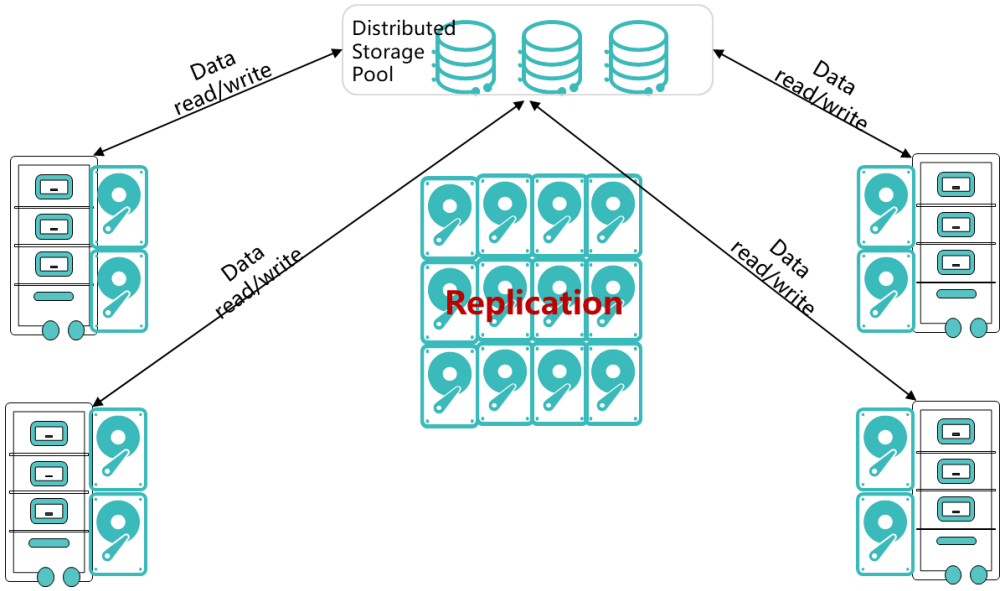


Figure 4-21 Distributed storage

As shown in Figure 4-21, the storage resources in a distributed storage system come from commodity x86 servers rather than dedicated storage devices. A distributed storage system has no controller or disk enclosures. The clients delivered by the distributed storage system are responsible for all of the following: identify and manage hard drives; establish routes; and execute I/Os.

The distributed storage client mode has both advantages and disadvantages.

In terms of capacity expansion, any x86 server with a client installed can be a part of the distributed system. Therefore, this mode delivers great scalability.

However, in addition to the applications running on the server, the client software installed on the server also consumes compute resources. When you plan a distributed storage system, you must reserve certain amounts of compute resources on servers you intend to add to this system. Therefore, this mode has certain requirements on the hardware resources of the server. In a traditional centralized storage system, data is read and written by controllers. The number of controllers is limited. In a distributed storage system, any servers with clients installed can read and write data. This improves the I/O speed to some extent because the distributed storage system does not have the controller bottleneck that exists in a centralized storage system. The paths for reading and writing data need to be calculated repeatedly. An

excessively large number of clients adds complexity to path calculation. This is why sometimes performance cannot be linearly improved simply by adding more clients.

To ensure high availability and security of data, the centralized storage system uses the RAID technology. RAID can be implemented by hardware and software. All hard drives in the same RAID array, regardless of software or hardware implementation, must reside on the same server (hardware RAID requires a unified RAID card, and software RAID requires a unified OS). Because the hard drives of a distributed storage system are distributed across different servers, the RAID mechanism simply cannot be used in such a system. Therefore, a replication mechanism is typically used in distributed storage systems to ensure high data reliability.

The replication mechanism keeps identical copies of data on different servers. The failure of a single server will not cause data loss. The distributed storage system combines local disks of all servers into several resource pools. Based on the resource pools, the distributed storage system delivers interfaces for creating and deleting application volumes and snapshots, and delivers volume device functions for upper-layer software, as shown in Figure 4-22.

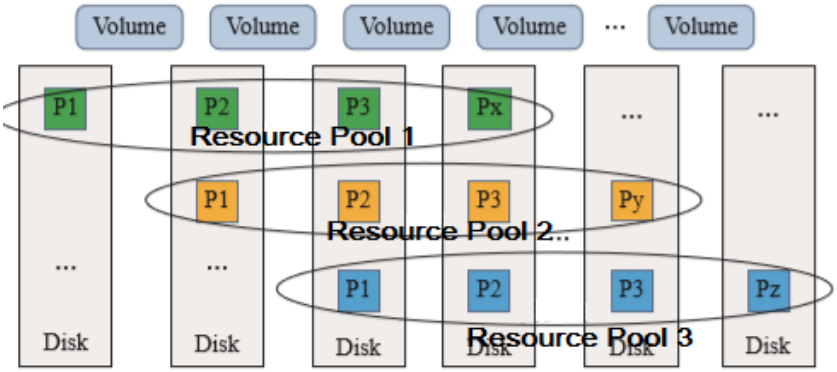


Figure 4-22 Distributed storage architecture

In a distributed storage system, each hard drive is divided into several partitions. Each partition belongs to only one resource pool. A partition functions as a data copy. The system ensures that multiple data copies are distributed on different servers (when the number of servers is greater than the number of data copies) and data consistency between multiple data copies. Then, data in the partitions is stored as key/value pairs.

The distributed storage system delivers volumes for the upper layer, which is easy to use. The system ensures that the number of active partitions is the same as that of standby partitions on each hard drive to avoid hot spots. All hard drives can be used as hot spares for resource pools. A resource pool supports up to hundreds of hard drives.

Figure 4-23 shows the main modules of the distributed storage architecture.

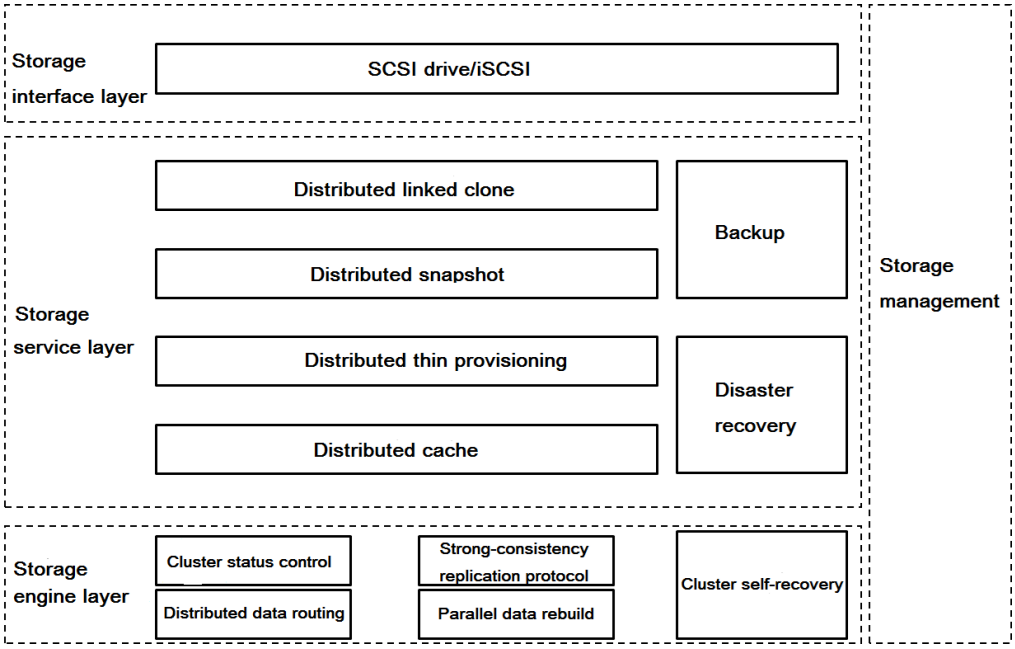


Figure 4-23 Distributed storage modules

Storage interface layer: delivers volumes for OSs and databases over the Small Computer System Interface (SCSI).

Storage service layer: delivers various advanced storage features, such as snapshot, linked cloning, thin provisioning, distributed cache, and backup and disaster recovery (DR).

Storage engine layer: delivers basic storage functions, including management status control, distributed data routing, strong-consistency replication, cluster self-recovery, and parallel data rebuilding.

Storage management layer: delivers the operation and maintenance (O&M) functions, such as software installation, automated configuration, online upgrade, alarm reporting, monitoring, and logging, and also delivers a portal for user operations.

When writing data, applications can use only the storage pool delivered by the distributed storage system. After the write requests of applications are sent to the storage pool, the data is copied to a specified number of copies (the number is manually set by users), and the write operation is delivered to different hard drives. The write operation is complete only after all messages indicating the write operation completion are returned.

When applications read data, they read data from the active copies rather than all copies. When the active copies are unavailable, the applications read data from other copies.

Common distributed storage products include Ceph (open source), HDFS, FusionStorage (Huawei), and vSAN (VMware).

Distributed storage has the following advantages:

- Excellent performance

  The distributed storage system uses an innovative architecture to organize SATA/SAS HDDs into a storage pool like SAN delivering a higher I/O than SAN devices and optimal performance. In a distributed storage system, SSDs can replace HDDs as high-speed storage devices, and InfiniBand networks can replace GE/10GE networks to deliver higher bandwidth, fulfilling the high performance requirements for real-time processing of large volumes of data.

The distributed storage system uses stateless software engines deployed on each node, eliminating the performance bottleneck of centralized engines. Moreover, these distributed engines deployed on standalone servers consume much fewer CPU resources and deliver higher IOPS than centrally-deployed engines do.

The system integrates computing and storage and evenly distributes cache and bandwidth to each server node. Each disk on the distributed storage system servers uses independent I/O bandwidths, preventing a large number of disks from competing for limited bandwidths between computing devices and storage devices in an independent storage system.

The distributed storage system uses some memory of each server for read cache and non-volatile dual in-line memory module (NVDIMM) for write cache. Caches are evenly distributed to all nodes. The total cache size on all servers is greatly larger than that delivered by external storage devices. Even if large-capacity and low-cost SATA hard drives are used, the distributed storage system can still deliver high I/O performance, improving the overall performance by 1 to 3 folds and providing larger effective capacity.

- Global load balancing

  The implementation mechanism of the distributed storage system ensures that I/O operations of upper-layer applications are evenly distributed on different hard drives of different servers, preventing partial hot spots and implementing global load balancing. The system automatically disperses data blocks on the hard disks of various servers. Data that is frequently or seldom accessed is evenly distributed on the servers, avoiding hot spots. FusionStorage employs the data fragment distribution algorithm to ensure that active and standby copies are evenly distributed to different hard disks of the servers. In this way, each hard disk contains the same number of active and standby copies. When the system capacity is expanded or reduced due to a node failure, the data reconstruction algorithm helps ensure load balancing among all nodes after system reconstruction.

- Distributed SSD storage

  The distributed storage system uses the SSD storage system that supports high-performance applications to deliver higher read/write performance than the traditional HDDs (SATA/SAS hard drives). PCIe SSDs deliver higher bandwidth and I/O. The PCIe 2.0 x8 interface provides a read/write bandwidth of up to 3.0 GB. I/O performance of SSDs delivers 4 KB random data transfer, realizing up to 600,000 continuous random read IOPS and 220,000 continuous random write IOPS. Although SSDs deliver high read and write speeds, SSDs have a shorter write lifespan. When SSDs are used, the distributed SSD storage system uses multiple mechanisms and measures to improve reliability.

- High-performance snapshot

  The distributed storage system delivers the snapshot mechanism, which allows the system to capture the status of the data written into a logical volume at a particular point in time. The data snapshot can then be exported and used for restoring the volume data when required. The snapshot data of the distributed storage system is based on the distributed hash table (DHT) mechanism. Therefore, the snapshots do not cause performance deterioration of the original volumes. The DHT technology delivers high query efficiency. For example, to build indexes for a 2 TB hard disk in the memory, tens of MB of memory space is required. Only one hash query operation can determine whether a snapshot has been created for the disk. If a snapshot has been created, the hash query can also determine the storage location of the snapshot.

- High-performance linked cloning

  The distributed storage system delivers the linked cloning mechanism for incremental snapshots so that multiple cloned volumes can be created for a snapshot. The data in the cloned volumes is the same as that in the snapshot. Subsequent modifications to a cloned

volume do not affect the snapshot or other cloned volumes. The distributed storage system supports batch deployment of VM volumes. Hundreds of VM volumes can be created in seconds. A cloned volume can be used for creating a snapshot, restoring data from the snapshot, and cloning the volume as the base volume again.

- High-speed InfiniBand (IB) network

  To eliminate storage switching bottlenecks in a distributed storage environment, a distributed storage system can be deployed on an IB network designed for high-bandwidth applications.

4.3 Virtualized Storage and Non-virtualized Storage

Storage virtualization described in this section refers to virtualization in a narrow sense. If a cluster has a file system, it is virtualized storage. Otherwise, it is non-virtualized storage. The file system can be an NFS or a virtual cluster file system. If no file system is available, the virtualized cluster needs to directly invoke logical volumes.

We know that physical disks reside at the bottom of the storage system, either centralized or distributed. After RAID or replication is implemented, physical volumes are created on top of these physical disks. In most cases, the physical volumes are not directly mounted to upper-layer applications, for example, OSs or virtualization systems (used in this document). The reason is that once a physical volume is mounted, all its space is formatted by upper-layer applications. After the storage space is used up, you can add disks to expand the capacity. However, you need to reformat the physical volume after the capacity expansion, which may cause data loss. Therefore, typically, multiple physical volumes are combined into a volume group, and then the volume group is virtualized into multiple logical volumes (LVs). The upper-layer applications use the spaces of the LVs.

In cloud computing, the virtualization program formats the LVs. Vendors use different virtual file systems. For example, VMware uses Virtual Machine File System (VMFS), and Huawei uses Virtual Image Manage System (VIMS). Both of them are high-performance cluster file systems that deliver a capacity exceeding the limit of a single system and allow multiple compute nodes to access an integrated clustered storage pool. The file system of a computing cluster ensures that no single server or application software have complete control over the access to the file system.

VIMS is used as an example. It is based on SAN storage. FusionStorage delivers only non-virtualized storage space. FusionCompute manages VM images and configuration files on VIMS. VIMS uses the distributed locking mechanism to ensure read/write consistency of cluster data. The minimum storage unit used by virtualization programs is logical unit number (LUN). LUNs correspond to volumes. Volumes are managed objects in the storage system. LUNs are external presentation of volumes. LUNs and volumes are allocated from the same resource pool.

After virtualization is used, LUNs can be divided as thick LUNs and thin LUNs.

As a traditional type of LUNs, thick LUNs support virtual resource allocation. They are easy to create, expand, and compress. A thick LUN gets full storage capacity from the storage pool once being created, namely, the LUN size equals to the allocated space. Therefore, the performance of a thick LUN is relatively high and predictable. Aside from thick LUNs, thin LUNs support virtual resource allocation. They are easy to create, expand, and compress.

An initial capacity allocation policy is set during the creation of thin LUNs. After thin LUNs are created, the storage system allocates an initial capacity to each LUN and retains the remaining capacity in the storage pool. When the usage of the allocated storage capacity reaches the threshold, the storage system allocates a certain amount of capacity from the

storage pool to the thin LUNs. This process repeats until the thin LUNs reach the preset full capacity. Therefore, thin LUNs have higher storage capacity utilization.

The differences between thick and thin LUNs are as follows:

- Capacity

  Thick LUNs, once being created, get full storage capacity from the storage pool.

  Thin LUNs get storage capacity on demand. A thin LUN is allocated with an initial capacity when created and then allocated with more capacity dynamically.

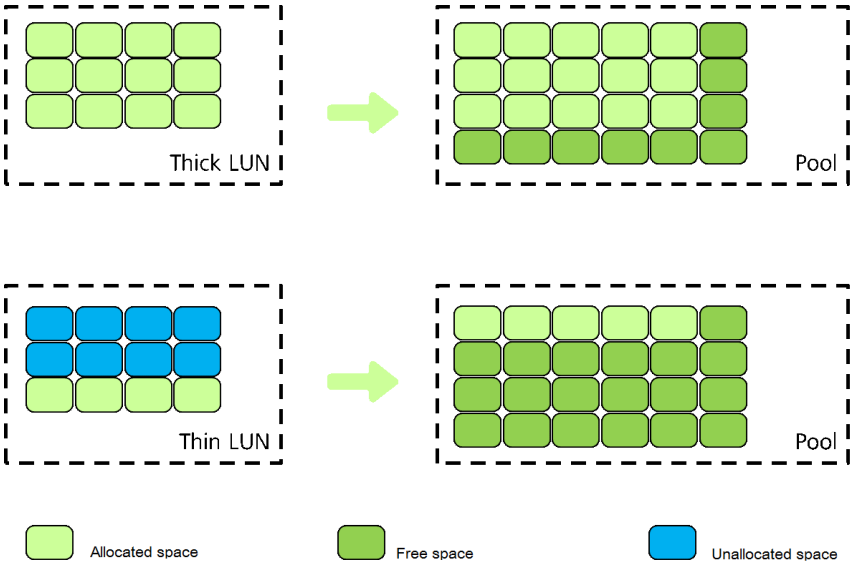


Figure 4-24 Thin LUNs

- Disk space reclamation

  Capacity reclamation here refers to releasing the capacity of some LUNs to the storage pool for the use of other LUNs. Capacity reclamation does not apply to a thick LUN, as it gets full capacity from the storage pool when created. Though data in a thick LUN is deleted, the allocated capacity is occupied by the thick LUN and cannot be used by other LUNs. However, if a thick LUN is deleted manually, its capacity can be reclaimed.

  When data in a thin LUN is deleted, space in the thin LUN can be released. In this way, storage capacity can be used dynamically, improving the utilization rate.

  Figure 4-25 shows the capacity reclamation of a thin LUN.

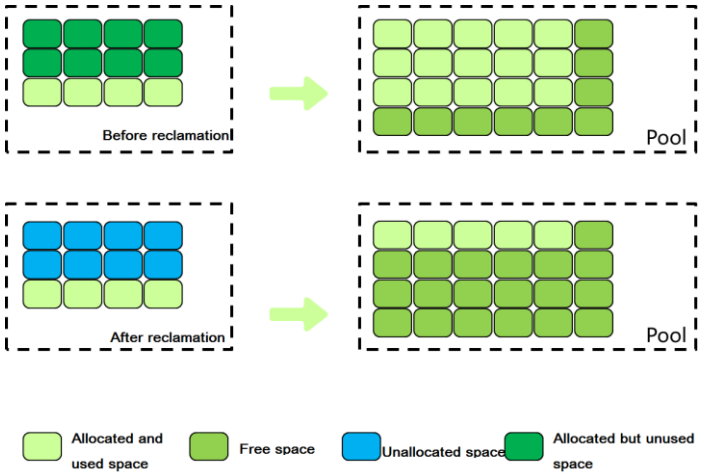


Figure 4-25 Disk space reclamation

- Performance

  A thick LUN delivers higher performance for sequential reads/writes as it gets full storage capacity from the beginning, but it has some storage capacity wasted.

  The performance of a thin LUN is hampered as background formatting is required each time the thin LUN expands capacity. In addition, capacity allocations may cause discontinuous disk storage space, so it takes more time for sequential reads/writes to find storage locations.

- Application scenarios

  Thick LUNs:

  - High performance is required.
  - Storage space utilization is less concerned.
  - Costs are insensitive.

  Thin LUNs:

  - Moderate performance is required.
  - Storage space utilization is more concerned.
  - Costs are sensitive.
  - Required storage capacity is hard to predict.

In addition to virtualized clustered file systems, common file systems include NAS systems (NFS and CIFS) and OS file systems.

A file system is a hierarchical structure of large numbers of files. An OS file system enables users to view data in the form of files and folders, and to copy, paste, delete, and restore data at any time. File systems use directories to organize data into hierarchical structures. Directories are the places where file pointers are stored. All file systems maintain this directory. The operating system maintains only the local directory. The cluster maintains the shared directory formed by the NAS or clustered file system.

The common OS file formats include FAT32 (Microsoft), NTFS (Microsoft), UFS (Unix), and EXT2/3/4 (Linux).

Figure 4-26 shows the working process of the OS file system.

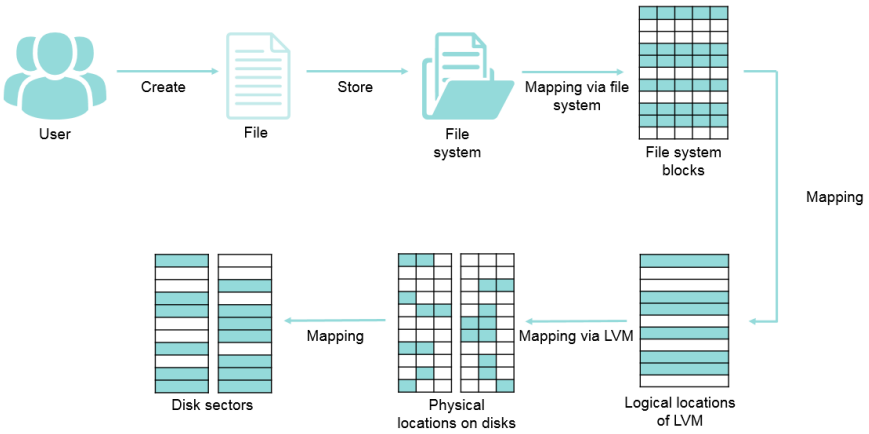


Figure 4-26 OS file system

A user or an application creates files or folders.

These files and folders are stored in the file system.

The file system maps data corresponding to these files to file system blocks.

The file system blocks correspond to the logical partitions formed by the logical volumes.

The logical partitions are mapped to the physical partitions of the physical disks by using the OS or LVM.

A physical partition contains one or more physical disks in a physical volume.

4.4 VM Disks

A VM consists of configuration files and disk files. Each VM disk corresponds to a disk file where user data is stored.

If virtualized storage is used, all disk files are stored in the shared directory of the file system. If non-virtualized storage is used, each disk file corresponds to a LUN. From the perspective of users and OSs, either files or LUNs are the same as common hard drives, which are displayed as hard drives among the hardware resources of the system. When creating a VM, the administrator needs to create disks for the VM to store data. The disk information corresponds to several lines in the configuration file.

Similar to other files, VM disk files have their own fixed formats. Table 4-4 lists common VM disk formats.

Table 4-4 Common VM disk formats

VM Disk File Format	Supported Vendor, Product, or Platform
RAW	All vendors
VMDK	VMware
VHD	Microsoft Hyper-V and Huawei FusionCompute
QCOW	QEMU or KVM virtualization platforms
QED	
VDI	Oracle

Each vendor can use its own tool to convert other VM disk formats to formats that can be used by its own products. For example, Huawei Rainbow can convert third-party or open-source VM disks to the VHD format.

4.5 Storage Features of Huawei Virtualization Products

4.5.1 Storage Architecture of Huawei Virtualization Products

FusionCompute can use the storage resources from dedicated storage devices or the local disks of hosts. Dedicated storage devices are connected to hosts through network cables or optical fibers.

FusionCompute uniformly converts storage resources into datastores. After datastores are associated with hosts, virtual disks can be created for VMs.

Storage resources that can be converted to datastores include:

- LUNs on SAN devices, including iSCSI storage devices and FC SAN storage devices
- File systems created on network attached storage (NAS) devices
- Storage pools on FusionStorage Block
- Local disks on hosts (virtualized)

In Huawei FusionCompute, these storage units are called storage devices, and physical storage media that deliver storage space for virtualization are called storage resources, as shown in Figure 4-27.

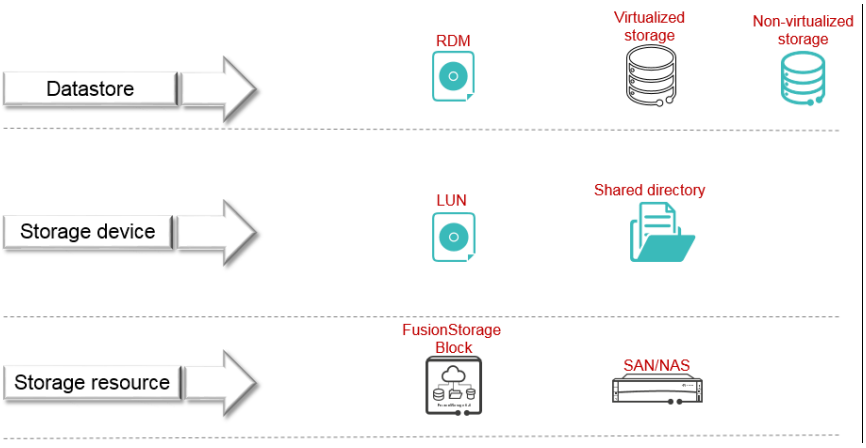


Figure 4-27 Huawei storage model

NOTE

When adding storage devices to FusionCompute, observe the Huawei-defined logical architecture and determine how devices at each logical layer are added to the system. For example, storage resources need to be manually added, and storage devices can be scanned.

Before using datastores, you need to manually add storage resources. If the storage resources are IP SAN, FusionStorage, or NAS storage, you need to add storage ports for hosts in the cluster and use the ports to communicate with the service ports of centralized storage controller or the management IP address of FusionStorage Manager. If the storage resources are provided by FC SAN, you do not need to add storage ports.

After adding storage resources, you need to scan for these storage devices on the FusionCompute portal to add them as datastores.

Datastores can be virtualized or non-virtualized. You can use LUNs as datastores and connect them to VMs from the SAN without creating virtual disks. This process is called raw device mapping (RDM). This technology applies to scenarios requiring large disk space, for example, database server construction. RDM can be used only for VMs that run certain OSs.

4.5.2 Characteristics of Huawei VM Disks

After adding datastores, you can create virtual disks for VMs. Customers may have various needs for using VMs, for example, they may want to share a VM disk to save more physical space. Therefore, Huawei VM disks are classified into different types based on these requirements.

- Based on sharing type, VM disks are classified as non-shared disks and shared disks.

  - **Non-shared**: A non-shared disk can be used only by a single VM.
  - **Shared**: A shared disk can be used by multiple VMs.

  If multiple VMs that use a shared disk write data into the disk at the same time, data may be lost. Therefore, you need to use application software to control disk access permissions.

- Based on the configuration mode, VM disks can be classified as common disks, thin provisioning disks, and thick provisioning lazy zeroed disks.
  - **Common**: The system allocates disk space based on the disk capacity. During disk creation in this mode, data remaining on the physical device will be zeroed out. The performance of the disks in this mode is better than that in the other two modes, but the creation duration may be longer than that required in the other modes.
  - **Thin provisioning**: In this mode, the system allocates part of the configured disk capacity for the first time, and allocates the rest disk capacity based on the storage usage of the disk until the configured disk capacity is allocated. In this mode, datastores can be overcommitted. It is recommended that the datastore overcommit rate not exceed 50%. For example, if the total capacity is 100 GB, the allocated capacity should be less than or equal to 150 GB. If the allocated capacity is greater than the actual capacity, the disk is in thin provisioning mode.
  - **Thick provisioning lazy zeroed**: The system allocates disk space based on the disk capacity. However, data remaining on the physical device is zeroed out only on first data write from the VM as required. In this mode, the disk creation speed is faster than that in the Common mode, and the I/O performance is between the Common and Thin provisioning modes. This configuration mode supports only virtualized local disks or virtualized SAN storage.
- Based on the configuration mode, VM disks can be classified as dependent disks, independent persistent disks, and independent non-persistent disks.
  - **Dependent**: A dependent disk is included in the snapshot. Changes are written to disks immediately and permanently.
  - **Independent persistent**: In this mode, disk changes are immediately and permanently written into the disk, which is not affected by snapshots.
  - **Independent non-persistent**: In this mode, disk changes are discarded after the VM is stopped or restored using a snapshot.

If you select **Independent persistent** or **Independent non-persistent**, the system does not take snapshots of the data on the disk when creating a snapshot for the VM. When the VM snapshot is used to restore a VM, the VM disks are not restored.

After a snapshot is taken for a VM, if disks on the VM are detached from the VM and not attached to any other VM, the disks will be attached to the VM after the VM is restored using the snapshot. However, data on the disks will not be restored.

If a disk is deleted after a snapshot is created for the VM, the disk will not be attached to the VM after the VM is restored using the snapshot.

Some disk types cannot be changed once they are set and some can be changed. For example, disk modes can be converted.

NOTE

For details about the characteristics of Huawei VM disks, see Scenario 1: VM Disk Application in the *Lab Guide*.

5 Virtualization Features

As we have discussed in previous sections, cloud computing is a service delivery model, and virtualization is a technology. Virtualization is the main technology in the Cloud Computing 1.0 era. So, what features does virtualization provide?

Before introducing virtualization features, let's go over the characteristics of virtualization again.

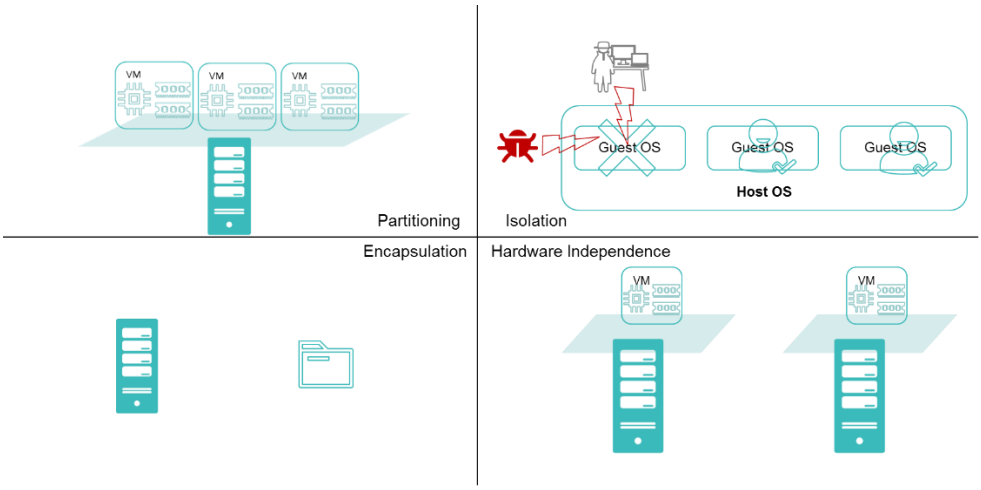


Figure 5-1 Virtualization characteristics

Based on these characteristics, we introduce virtualization features from the aspects of clusters and VMs.

5.1 Virtualization Cluster Features

A cluster is a group of computers that collectively provide resources for users, including compute, storage, and network. A complete cluster must contain all these types of resources.

5.1.1 HA

Let's begin with a story about Kai-Fu Lee, a venture capitalist, technology executive, write, and an artificial intelligence (AI) expert currently based in Beijing, China.

In the early years, Dr. Kai-Fu Lee worked in Apple, specializing in the development of new products.

At one time, Kai-Fu Lee and the CEO of the company, Mr. Sculley, were invited by America's most popular morning TV show, "Good Morning America." The TV station communicated with Apple in advance and hoped that they could demonstrate Apple's latest speech recognition system on live TV.

However, the success rate of the system was about 90%, and Mr. Sculley hoped to increase the success rate to 99%.

Kai-Fu Lee did not modify the program of the system, but prepared two identical computers. If one computer becomes faulty, the system will be immediately switched over to the other computer. This way, there is just a possibility of 1% (10% x 10%) that both the computers become faulty, and the success rate of the system reaches 99%.

This story shows us the basic principles of HA. The cluster technology is used to overcome the limitations of a single physical host, thereby preventing service interruption or reducing system downtime. HA in virtualization is actually the compute-layer HA, that is, VM HA. If a compute node becomes faulty, the other compute node in the cluster can be automatically started.

A virtualization cluster usually uses shared storage. A VM consists of configuration files and data disks. The data disks are stored on shared storage, and the configuration files are stored on the compute nodes. If a compute node becomes faulty, the virtualization management system (such as vCenter and VRM) rebuilds the fault VM on other nodes based on the recorded VM configuration information.

During HA, the following issues need to be addressed:

- Detecting host faults
- Handling VM startup failures

First, let's look at the first issue. To check whether a compute node is faulty, the administrator needs to periodically establish communication with all nodes in the cluster. If the communication with a node fails, the node may be faulty. Use Huawei FusionCompute as an example. The heartbeat mechanism between the CNA host and VRM enables VRM to check whether CNA is faulty. The detailed process is as follows:

1. The heartbeat mechanism runs in a process or service on the CNA host.
2. The host sends heartbeat messages to VRM at an interval of 3s. If the host does not send heartbeat messages to VRM within 30s for ten consecutive times, the host is set to the faulty state. In this case, the "Heartbeat Communication Between the Host and VRM Interrupted" alarm is generated on the FusionCompute portal.
3. A timeout mechanism is used when the host reports heartbeat messages to VRM. The timeout interval for socket connecting, receiving, and sending is 10s. If the VRM service or network is abnormal, timeout may occur. The timeout log is printed with a 13-second timestamp (Timeout detection interval 3s + Socket timeout period 10s = 13s).
4. After receiving a heartbeat message from a host, VRM sets the heartBeatFreq variable to 10 (the default value is 10, which can be changed by modifying the configuration file). The detection thread decreases the value by 1 every 3s and checks the current value of this parameter. If the value is less than or equal to 0, the system regards that the host is

abnormal, reports an alarm on the FusionCompute portal, and sends a host exception message to VRM. VRM then determines whether to trigger VM HA.

Next, let's look at the second issue. When a VM is started on another host, services on the VM may fail to automatically start, and even the OS may fail to start. Service recovery on the VM may fail or take a long time. In this case, HA on the service plane is needed. If the active VM is faulty or cannot be restored, services will be recovered on the standby VM using HA technologies such as the floating IP address and Keepalived.

Therefore, VM HA is usually used together with application-layer HA to improve the service recovery success rate and efficiency.

In addition to the preceding issues, how to prevent split-brain needs to be considered. Split-brain may occur when the shared storage is written by two VMs at the same time. Before HA is implemented, the management system uses the heartbeat mechanism to detect whether the compute node is faulty. If only the heartbeat network is faulty, the management system may incorrectly determine that the compute node is faulty. In this case, split-brain may occur. Therefore, before starting a VM, the system checks whether data is written to the storage device. If data is written to the storage device, the host may not be faulty. In this case, the system does not start the VM. Instead, a message indicating that HA is successful is displayed on the management system page.

If a node in the cluster becomes faulty, VMs on this node are automatically migrated to an alternative host that is running properly.

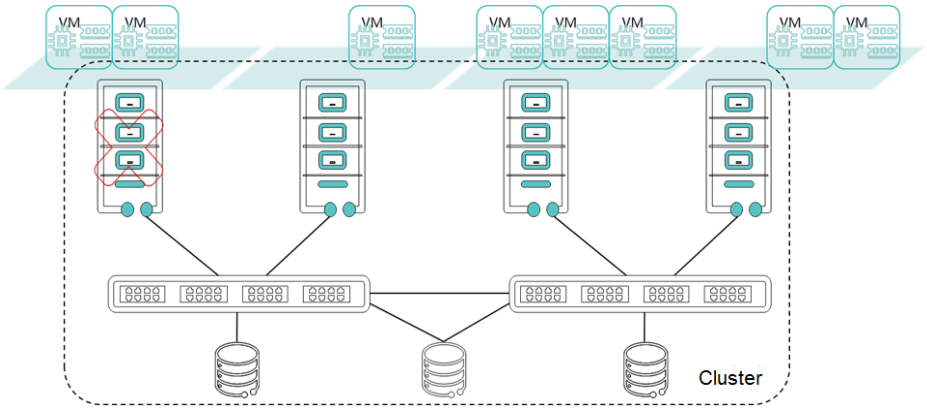


Figure 5-2 HA diagram

5.1.2 Load Balancing

Load balancing is a cluster technology that distributes loads (such as network services and traffic) to multiple network devices (such as servers and firewalls) or links. This improves service processing capabilities and ensures high service reliability.

Load balancing has the following advantages:

- High performance: Load balancing evenly distributes loads to multiple devices, improving the performance of the entire system.
- Scalability: Load balancing easily increases the number of devices or links in a cluster to meet the service growth requirements without degrading the service quality.
- High reliability: If one or more devices or links are faulty, services are not interrupted, improving system reliability.

- Easy management: Management work is concentrated on the devices that use the load balancing technology. The device group or link group only needs to be configured and maintained.
- Transparency: For users, a cluster is a device or link with high reliability and performance. Users do not need to care about the network structure. Adding or deleting devices or links does not affect services.

In a virtualization environment, load balancing is generally implemented on compute nodes based on the node CPU and memory usages. During VM creation and running, the management system detects the usage of all physical resources in the cluster, uses the intelligent scheduling algorithm to determine the optimal host for running VMs, and migrates VMs to the optimal host by means of live migration, thereby improving global service experience.

As shown in Figure 5-3, there are four compute nodes in the cluster and eight VMs with the same specifications are running. When the administrator creates a VM and does not specify the host for the VM, the system automatically creates the VM on the second or fourth node with light load. A large number of VMs are running on the third host. After a period of time, the system automatically detects that the load on the third host is heavy and migrates VMs on this host to other light-load hosts.

The load threshold can be specified by the system administrator or defined by the system. In Huawei FusionCompute, if the CPU and memory usage of a host exceeds 60%, VRM migrates VMs on the host to other hosts. Before the migration, the administrator can set an automatic migration task. Alternatively, the administrator can manually perform the migration after receiving a notification.

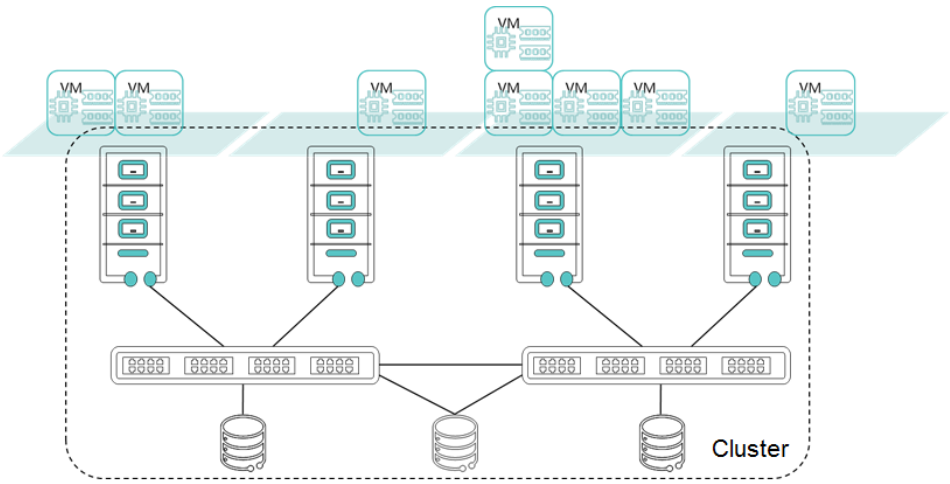


Figure 5-3 Load balancing diagram

5.1.3 Scalability

In a traditional non-virtualization environment, all services are deployed on physical machines. In the initial phase of system construction, the service volume is not large. Therefore, the hardware resources configured for physical machines are limited. As the service volume increases, the original hardware cannot meet service requirements and needs to be upgraded continuously, for example, adding CPUs or expanding memory from 256 GB to 512 GB. This scaling mode is called vertical scaling, or scale-up. However, the hardware resources supported by a physical machine have an upper limit. If the service volume keeps increasing, the physical machine must be replaced, and the service migration requires machine shutdown.

In virtualization, all resources are pooled to carry service VMs. When the service volume increases, you only need to add resources to the resource pool, without having to add hardware resources. During the actual implementation, you only need to add servers. This scaling mode is called horizontal scaling or scale-out.

Clusters support horizontal scaling, which is easier than the scaling in traditional non-virtualized IT systems. As shown in Figure 5-4, the original cluster contains only four servers and three storage devices. At the early stage of construction, the resources in the resource pool are sufficient. After some resources are consumed by VMs, the administrator finds that the existing resources are insufficient to support new services. The administrator can migrate all VMs to certain physical servers in a cluster during off-peak hours and power off and upgrade other hardware resources. The administrator can also add servers and storage devices to a cluster without affecting services. Scale-up or scale-out in virtualization systems is easier to implement than scaling in non-virtualization systems and has less impact on services.

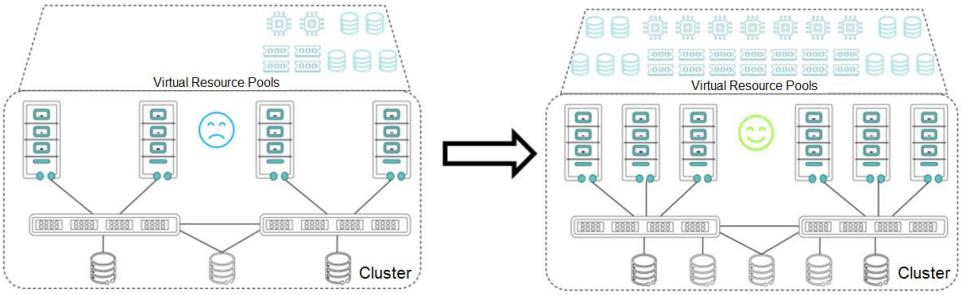


Figure 5-4 Resource hot-add

5.1.4 Memory Overcommitment

Memory overcommitment allows VMs to use more memory than the total physical memory of the server by leveraging specific technologies to improve VM density.

Memory overcommitment technologies include memory ballooning, memory swapping, and memory sharing. Generally, the three technologies need to be applied together and take effect at the same time.

Memory sharing: Multiple VMs share the memory page on which the data content is the same. See Figure 5-5 for details. In this figure, the physical host provides 4 GB physical memory for the hypervisor and allocates the memory to three VMs. The three VMs read data from the same physical memory. According to the memory virtualization implementation principles, the hypervisor maps this memory segment to different VMs. To prevent any VM from modifying data in this memory segment, all VMs have only the read permission on this memory segment. If VMs need to write data to this memory segment, the hypervisor needs to create a memory segment for mapping. With the memory sharing technology, 6 GB of virtual memory can be allocated to VMs based on 4 GB of physical memory.

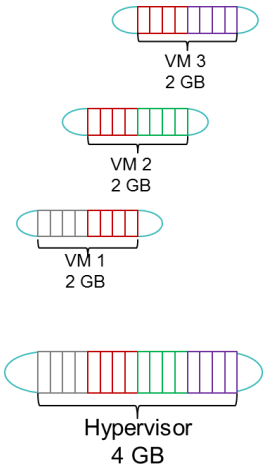


Figure 5-5 Memory overcommitment

Memory ballooning: The system automatically reclaims the unused memory from a VM and allocates it to other VMs to use. Applications on the VMs are not aware of memory reclamation and allocation. The total amount of the memory used by all VMs on a physical server cannot exceed the physical memory of the server. See Figure 5-6 for details.

Figure 5-6 Memory ballooning

Each of the three VMs has 2 GB virtual memory. The memory usage of VM 1 is only 20%, and the memory usages of VM 2 and VM 3 are 70%. The system automatically maps physical memory allocated to VM 1 to VM 2 and VM 3 in the background to relieve the memory pressure.

Memory swapping: External storage is virtualized into memory for VMs to use. Data that is not used temporarily is stored to external storage. If the data needs to be used, it is exchanged with the data reserved on the memory. See Figure 5-7 for details.

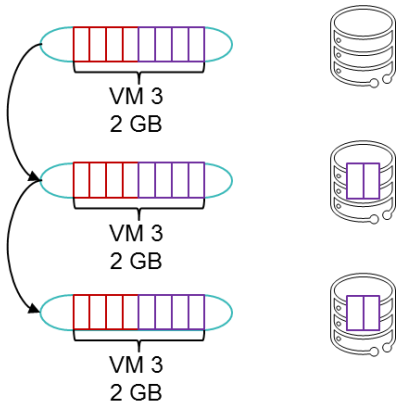


Figure 5-7 Memory swapping

Memory swapping is similar to the functions of the virtual memory of Windows and the swap partition of Linux. They simulate storage as memory to store the data that has been called to the memory but is seldom used on disks. When the data is used, the data will be called back to the memory.

In some virtualization products, such as FusionCompute, memory overcommitment is configured based on clusters. After memory overcommitment is enabled, the memory overcommitment policy replaces the physical memory allocation policy. When the memory is sufficient, VMs can use all physical memory. If the memory is insufficient, the system schedules memory resources based on the memory overcommitment policies using memory overcommitment technologies to release free memory.

Memory overcommitment reduces customers' costs.

The feature helps increase VM density when the memory size of compute nodes is fixed.

The feature eliminates the need for storage devices when the VM density of compute nodes is fixed.

5.2 VM Features

5.2.1 Quick VM Deployment

When we purchase an Elastic Cloud Server (ECS) on the public cloud, a VM with an OS will be generated in the background. The time required for installing an OS on the VM is much shorter than that required for installing an OS on a PC. This is the quick VM deployment feature.

VMs can be quickly deployed by using templates or by cloning other VMs.

A VM template is a copy of a VM. It contains VM disks and VM configuration files. Creating a VM using a VM template can greatly reduce the time for configuring a new VM and installing an OS. After a VM template is created, it cannot be started or modified. This ensures that the template will not be modified by other users and does not occupy compute resources of the cluster. A VM deployed using a template is independent of the template. To update or edit the template, you need to convert the template to a VM first. After the template is edited, you can convert the VM into a template.

VM templates are useful for deploying a large number of similar VMs because it can keep VM consistency and parameters can be modified automatically (such as the host name and SID) as required. For example, if a group of test personnel need to test the newly developed software, the administrator can use a VM with the software installed as a template, quickly deploy a batch of VMs with the same configuration, and assign the VMs to different test personnel for testing in different scenarios and with different requirements. If any problem occurs during the test, the administrator can delete the faulty VM and deploy a new VM for the test personnel. In addition, different VM templates may include different software. For example, a financial system may be pre-installed in a VM template used by financial personnel, and an ERP system may be pre-installed in a VM template used by sales personnel. VMs can be created for different departments using different templates.

In addition to deploying a VM using a template, you can also quickly deploy a VM using another VM. This function is called VM cloning. Different from template-based deployment, VM cloning is a process of completely copying the source VM at a specific time point. All settings of each cloned VM, including the host name and IP address, are the same as those of the source VM. The system reports an error if there are two identical IP addresses in a LAN. Therefore, the cloned VM and the source VM cannot be started at the same time.

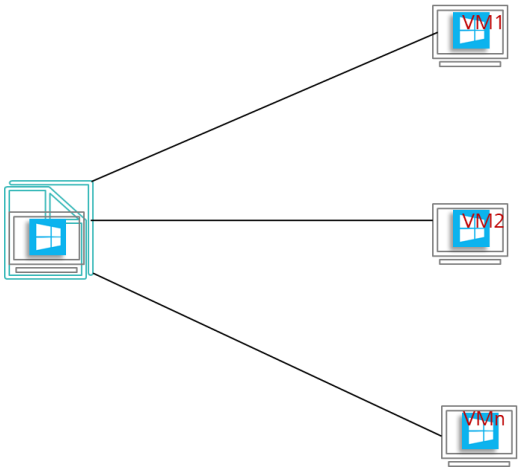


Figure 5-8 Deploying a VM using a template

5.2.2 VM Resource Hot-Add

Hot-add is to add compute, storage, and network resources to a VM when the VM is started.

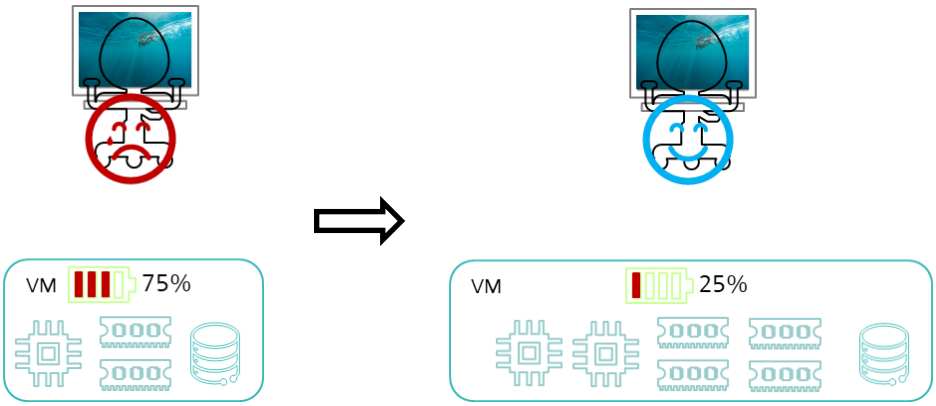


Figure 5-9 VM resource hot-add

From the administrator's perspective, the CPU and memory parameters of a VM are part of the VM configuration file. The hardware configuration of the VM can be modified through modification of these parameters. As shown in Figure 5-9, when a user is using a VM, the CPU and memory usage reaches 75%, which may affect user experience or even affect the normal use of the VM. In this case, the VM resource hot-add feature can be used to add CPU and memory resources to the VM online. This feature allows the resource usage to be quickly reduced to the normal level.

In addition to CPU and memory resources, storage and network resources also support hot-add. For example, the disk capacity of a VM can be expanded or a NIC can be added to a VM.

The hot-add function needs to be supported by the VM and OS so that the added resources can take effect immediately. Otherwise, the VM must be restarted and can be used only after the OS identifies the hardware resources.

In most cases, resources support hot-add but do not support hot-delete. For example, the administrator can expand the capacity of a disk from 10 GB to 20 GB but cannot necessarily reduce the disk capacity from 20 GB to 10 GB.

In terms of storage resource addition, you can expand the capacity of existing disks and add disks to VMs.

5.2.3 VM Console

Physical machines can be operated using monitors but VMs cannot. Once the network is disconnected, VMs may be controllable. A real-time VM control technology is required.

Most hardware devices that cannot be connected to the display system are configured with an independent management interface for users to perform management operations. For example, in addition to service interfaces, storage devices are configured with control interfaces, and network devices are configured with console interfaces. Can VMs be controlled in the same way? The answer is yes.

Figure 5-10 Remote control over VMs

Each VM vendor provides the console management function for VMs. For example, VMware uses Remote Console, and Huawei and KVM use VNC. The console is network-dependent. If you need to log in to a VM from the console, you do not need to configure an IP address for the VM but must configure an IP address for the compute node where the VM is located. In

addition, the compute node functions as the server to provide services for users to log in to the VM. Therefore, the network between the client and the server must be normal.

5.2.4 VM Snapshot

In virtualization, VM snapshots are similar to pictures we take in our life. A snapshot records the VM status at a certain moment and contains complete data of the VM. You can use a snapshot to restore a VM to the state at a specific time point.

Generally, VM snapshots are used to quickly restore a VM if the VM is faulty before an upgrade, patch installation, or test is performed on the VM. The VM snapshot function is implemented by the storage system. Storage Networking Industry Association (SNIA) defines a snapshot as a fully usable copy of a defined collection of data that contains an image of the data as it appeared at the point in time at which the copy was initiated. A snapshot can be either a copy or a replication of the specified data. Figure 5-11 shows the diagram of the snapshot technology.

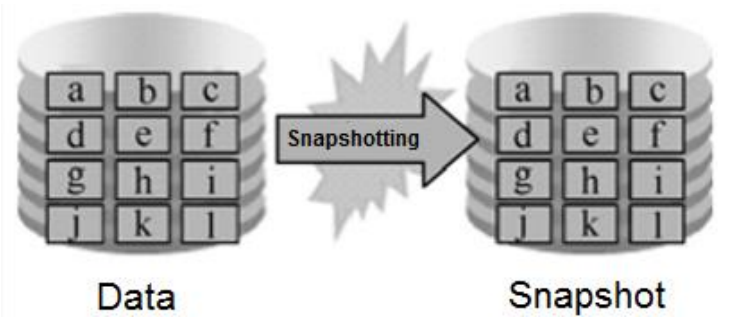


Figure 5-11 Snapshot

The snapshot technology has the following features:

Snapshots can be quickly generated and used as data sources for traditional backup and archiving, reducing or eliminating data backup windows.

Snapshots are stored on disks and can be quickly accessed, accelerating data restoration.

Disk-based snapshots provide flexible and frequent restoration points for storage devices. Snapshots at different points in time can be used to restore data that is accidentally erased or damaged online.

In terms of specific technical details, a snapshot creates a pointer list that indicates an address for reading data. When data changes, the pointer list can provide real-time data in a very short time and perform replication.

Common snapshot modes are Copy-On-Write (COW) and Redirect-On-Write (ROW). Both COW and ROW are concepts in the storage domain. Most vendors use the ROW technology to create VM snapshots. No matter whether COW or ROW is used, physical copy operations will not be performed, and only the mapping is modified. The following describes the differences between COW and ROW.

- COW

  Data is recorded in data blocks. When COW is used, the system generates a new space each time a snapshot is created. Once data in a data block changes, the system copies data in the original data block to a new space and then writes new data to the original data block. The copy operation is performed before data is written. For example, there is a parking lot, and cars are parked in parking spaces. A new car (car B) can be parked in only after a car (car A) in a parking space has been moved to another parking space. If

you were the owner of car A, would you feel that there was a problem with the management of the parking lot when the parking lot administrator told you to make room for other cars? Why not let the new car park into another parking space? ROW is introduced to address this issue.

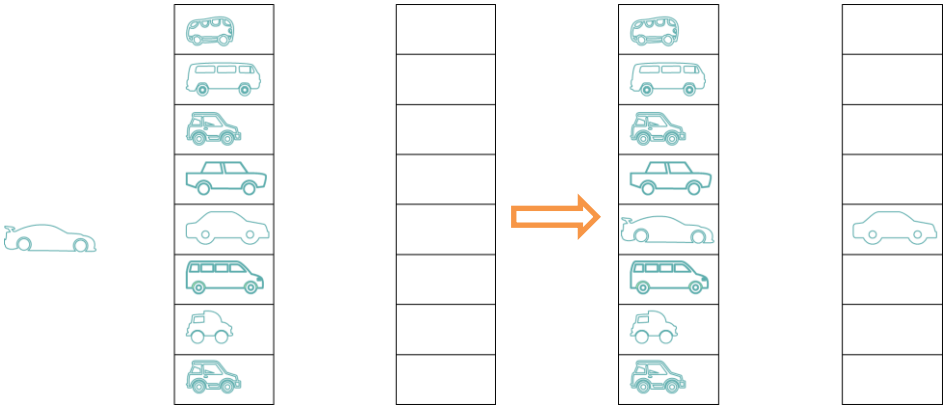


Figure 5-12 COW diagram

- ROW

  Similar to COW, data is recorded in data blocks and a new space is generated when a snapshot is created. The difference from COW is as follows: if new data is written into the system, the original data remains unchanged and new data is written to a new space. Let's go back to the previous example. If a new car is parked in, the administrator can direct the car to a new parking space.

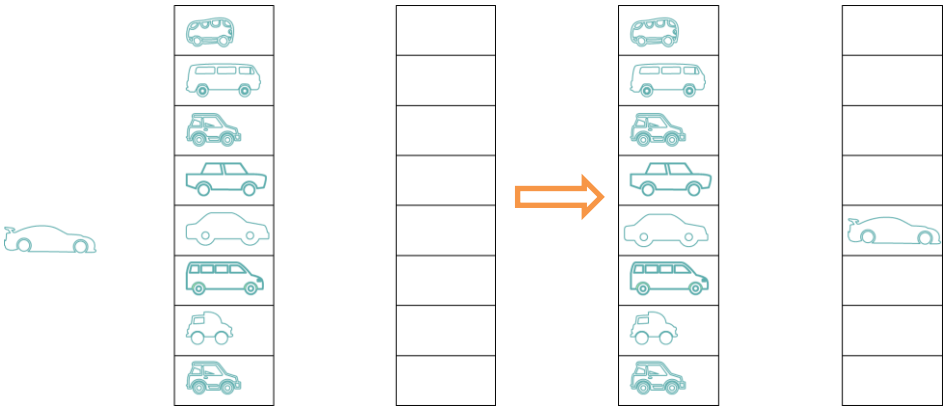


Figure 5-13 ROW diagram

COW writes data twice, whereas ROW writes data only once. Therefore, ROW is more advantageous than COW in terms of creating snapshots for VMs.

Multiple snapshots can be created for a VM to form a snapshot chain. Operations on any snapshot do not affect other snapshots.

5.2.5 NUMA

Non Uniform Memory Access (NUMA) is a technology that can improve the data read/write speed.

In modern times, the computing speed of a single CPU has reached the bottleneck, so designers adopt multi-core CPUs to improve the computing speed of computers. The CPU and memory are connected through the northbridge. As the number of CPUs increases, the

memory increases accordingly. As a result, the response speed on the northbridge becomes slower and slower. Therefore, designers evenly bind the memory to each CPU to avoid congestion caused by northbridge sharing. Figure 5-14 shows the comparison in multicore CPU ordering.

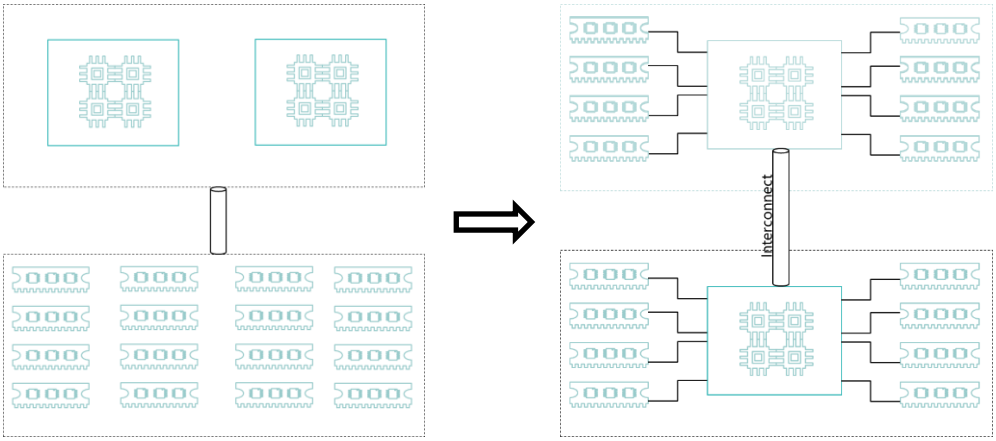


Figure 5-14 Comparison in multicore CPU ordering

After the modification, memory is associated with CPUs. A CPU can access its own local memory faster than non-local memory. As the local access speed is high, it is more efficient for a program to only use a CPU and its associated local memory. This technology is called NUMA.

NUMA nodes are CPU/memory couples. Each node has CPUs, bus, and memory. Cross-node access requires NUMA interconnects between the CPUs.

In virtualization scenarios, NUMA enables VMs to use hardware resources on the same NUMA node, improving the VM response speed.

5.3 Huawei Virtualization Product Features

As an advanced virtualization product, Huawei FusionCompute only supports the cluster and VM features aforementioned but also provides enhancements to general virtualization features. To use all features supported by FusionCompute, you need to install the Tools on VMs.

The Tools consists of the kernel-mode hardware driver and user-mode vm-agent process. The kernel-mode hardware driver allows administrators to perform VM operations such as snapshot creation, live migration, online VM flavor adjustment, and NIC QoS setting. The user-mode vm-agent process enables VRM to obtain the IP address and status of the VMs as well as soft power off and restart on VMs.

Tools installation methods vary depending on operating systems. For details about how to install Tools, see the product documentation. After Tools is installed, uninstall the Tools image. Otherwise, live VM migration and VM HA will be affected.

5.3.1 Cluster Features

Huawei FusionCompute supports memory overcommitment and NUMA. They are common virtualization technologies and therefore are not described in detail.

5.3.1.1 HA

In FusionCompute, clusters support multiple HA policies. Users can configure HA policies as required.

- **Host Fault Policy**
  - **Restore VM on Original Host**: When the host accommodating the VM is faulty, the system restarts the VM only after the host is restored.
  - **HA**: When the host accommodating the VM is faulty, the system restarts the VM from another host based on the VM startup policy configured for the cluster.
- **Datastore Fault Policy**
  - **Datastore Fault Handling by Host**: You are advised to select **No action**. If this parameter is set to **No action** and the datastore fault duration exceeds the value of **Policy Delay**, the VM receives IO ERROR, causing the OS to become abnormal. Alternatively, you can select **HA**. If the datastore fault duration exceeds the value of **Policy Delay**, the system starts the VM from another host based on the VM startup policy configured for the cluster.
  - **Policy Delay** (minutes): Indicates the delay for host handling policy upon datastore faults to take effect. Within this duration, the backend driver resends VM I/O requests. You are advised to set **Policy Delay** to a larger value that does not exceed 14400.
- **VM Fault Policy**
  - No action
  - Restart VM
  - HA

NOTE

In FusionCompute 6.3, only the Windows VM blue screen of death (BSOD) handling policy is supported for VM faults.

5.3.1.2 DPM

Distributed Power Management (DPM) enables the system to periodically check resource usage on hosts in a cluster. If resources in the cluster are sufficient but service load on each host is light, the system migrates VMs to other hosts and powers off these hosts to reduce power consumption. If the in-service hosts are overloaded, the system powers on offline hosts in the cluster to balance load among hosts.

When the automated power management function is enabled, automatic resource scheduling must also be enabled so that the system automatically balances VM load on hosts after the hosts are powered on.

The time-based power management settings allow the system to manage power in different time periods based on service requirements. When services are running stably, set automated power management to a low level to prevent adverse impact on services.

With the automated power management function enabled, the system checks resource usage in the cluster, and powers off some light loaded hosts only when the resource utilization drops below the light-load threshold over the specified time period (40 minutes by default). Similarly, the system powers on some hosts only when the resource utilization rises above the heavy-load threshold over the specified time period (5 minutes by default). You can customize the time period for evaluating the threshold of powering on or off hosts.

DPM can be used to meet energy-saving and environmental protection requirements of enterprises. For example, if an enterprise uses the FusionCompute-based desktop cloud, all VMs used in the cloud can be migrated to certain physical hosts at 22:00 p.m. in the evening,

and other physical hosts can be powered off. At 07:00 a.m. the next day, the system powers on all physical hosts and evenly distributes VMs to the physical hosts based on load balancing principles. When the physical hosts are powered off in the evening, auxiliary devices, such as the air conditioning and fresh air systems, can also be powered off to minimize energy consumption.

5.3.1.3 DRS Rules

Distributed Resource Scheduling (DRS) and DPM are part of load balancing. DRS provides a reference for system migration during load balancing.

- **Keep VMs together**: This rule keeps the selected VMs always running on the same host. One VM can be added to only one keep-VMs-together rule group.
- **Keep VMs mutually exclusive**: This rule keeps the selected VMs running on different hosts. One VM can be added to only one VM-mutually-exclusive rule group.
- **VMs to hosts**: This rule associates a VM group with a host group so that the VMs in the specified VM group can be configured to run only on the specified host in the host group.

If different rules conflict, the scheduling priorities of the rules are as follows:

- Highest priority: The rule type is **VMs to hosts**, and the rules are **Must run on hosts in group** and **Must not run on hosts in group**.
- Second priority: The rule types are **Keep VMs together** or **Keep VMs mutually exclusive**.
- Lowest priority: The rule type is **VMs to hosts**, and the rules are **Should run on host group** and **Should not run on hosts in group**.

5.3.1.4 IMC

On FusionCompute, the incompatible migration cluster (IMC) feature can be configured for a cluster to enable VM migration across hosts that use CPUs of different performance baselines in the cluster.

Currently, IMC is available for the CPUs from Intel only.

IMC allows VMs running on the hosts in the cluster to present the same CPU features. This ensures successful VM migration across these hosts even if the hosts use physical CPUs of different performance baselines.

IMC can be enabled for a cluster that contains hosts and VMs only when the following conditions are met:

- The CPU generations of the hosts in the cluster are the same as or later than the target IMC mode.
- The CPU generations of the running or hibernated VMs in the cluster are the same as or earlier than the target IMC mode. If any VM in the cluster does not meet this requirement, the VM must be stopped or migrated to another cluster.

5.3.2 VM Features

5.3.2.1 VM Resource QoS

CPU QoS

The CPU QoS ensures optimal allocation of compute resources for VMs and prevents resource contention between VMs due to different service requirements. It effectively increases resource utilization and reduces costs.

During creation of VMs, the CPU QoS is specified based on the services to be deployed. The CPU QoS determines VM computing capabilities. The system ensures the CPU QoS of VMs by ensuring the minimum compute resources and resource allocation priority.

CPU QoS is determined by the following aspects:

- CPU Quota

  CPU quota defines the proportion based on which CPU resources to be allocated to each VM when multiple VMs compete for the physical CPU resources.

  For example, three VMs (A, B, and C) run on the host that uses a single-core physical CPU with 2.8 GHz frequency, and their quotas are set to 1000, 2000, and 4000, respectively. When the CPU workloads of the VMs are heavy, the system allocates CPU resources to the VMs based on the CPU quotas. Therefore, VM A with 1000 CPU quota can obtain a computing capability of 400 MHz. VM B with 2000 CPU quota can obtain a computing capability of 800 MHz. VM C with 4000 CPU quota can obtain a computing capability of 1600 MHz. (This example explains the concept of CPU quota and the actual situations are more complex.)

  The CPU quota takes effect only when resource contention occurs among VMs. If the CPU resources are sufficient, a VM can exclusively use physical CPU resources on the host if required. For example, if VMs B and C are idle, VM A can obtain all of the 2.8 GHz computing capability.

- CPU Reservation

  CPU reservation defines the minimum CPU resources to be allocated to each VM when multiple VMs compete for physical CPU resources.

  If the computing capability calculated based on the CPU quota of a VM is less than the CPU reservation value, the system allocates the computing capability to the VM according to the CPU reservation value. The offset between the computing capability calculated based on the CPU quota and the CPU reservation value is deducted from computing capabilities of other VMs based on their CPU quotas and is added to the VM.

  If the computing capability calculated based on the CPU quota of a VM is greater than the CPU reservation value, the system allocates the computing capability to the VM according to the CPU quota.

  For example, three VMs (A, B, and C) run on the host that uses a single-core physical CPU with 2.8 GHz frequency, their quotas are set to 1000, 2000, and 4000, respectively, and their CPU reservation values are set to 700 MHz, 0 MHz, and 0 MHz, respectively. When the CPU workloads of the three VMs are heavy:

  - According to the VM A CPU quota, VM A should have obtained a computing capability of 400 MHz. However, its CPU reservation value is greater than 400 MHz. Therefore, VM A obtains a computing capability of 700 MHz according to its CPU reservation value.
  - The system deducts the offset (700 MHz minus 400 MHz) from VMs B and C based on their CPU quotas.

  - VM B obtains a computing capability of 700 (800 minus 100) MHz, and VM C obtains a computing capability of 1400 (1600 minus 200) MHz.

  The CPU reservation takes effect only when resource contention occurs among VMs. If the CPU resources are sufficient, a VM can exclusively use physical CPU resources on the host if required. For example, if VMs B and C are idle, VM A can obtain all the 2.8 GHz computing capability.

- CPU Limit

  CPU limit defines the upper limit of physical CPUs that can be used by a VM. For example, if a VM with two virtual CPUs has a CPU limit of 3 GHz, each virtual CPU of the VM can obtain a maximum of 1.5 GHz compute resources.

Memory QoS

Memory QoS allows VM memory to be intelligently reused based on the preset percentage of reserved memory. Memory overcommitment technologies, such as memory ballooning, are used to provide more virtual memory resources for VMs. This function maximizes memory resource reuse, increases resource utilization, ensures that a VM can obtain the reserved memory at least, and guarantees reliable service operating.

System administrators can set the reserved memory percentage based on service requirements. The main principle of memory overcommitment is to first use the physical memory.

Memory QoS is determined by the following aspects:

- Memory Quota

  Memory quota defines the proportion based on which memory resources to be allocated to each VM when multiple VMs compete for physical memory resources.

  The system allocates memory resources to VMs based on the proportion when VMs request memory resources or hosts release free memory resources (such as when VMs are migrated or stopped).

  CPU resources can be scheduled in real time. Memory resources are scheduled subtly and continuously when VMs are running until the configured VM memory resources are allocated to VMs.

  For example, three VMs each with 4 GB memory run on a host that has 6 GB memory, and their memory quotas are set to 20480, 20480, and 40960, respectively. In this case, the memory allocation ratio is 1:1:2. When the memory workloads on the three VMs gradually increase, the system subtly adjusts memory resources on the VMs based on the memory quotas until the VMs obtain 1.5 GB, 1.5 GB, and 3 GB memory, respectively.

  The memory quota takes effect only when resource contention occurs among VMs. If the memory resources are sufficient, a VM can exclusively use memory resources on the host if required. For example, if the memory resources required by VMs B and C are less than the reserved memory values, and VM A has more memory workloads to handle, VM A can use memory resources from free memory resources and memory resources on VMs B and C, until the memory resources obtained by VM A reach the upper limit, or the free memory resources are used up and memory resources on VMs B and C drop to the reserved values. For example, if VM C is not under memory pressure and has 1 GB memory reserved, VMs A and B theoretically can obtain a maximum of 2.5 GB memory resources each.

- Memory Reservation

  Memory reservation defines the minimum memory resources to be allocated to each VM when multiple VMs compete for memory resources.

A VM exclusively uses its reserved memory, that is, if certain memory resources are reserved for a VM, other VMs cannot use the memory resources even if the memory resources are idle.

- Memory Limit

  Memory limit specifies the maximum physical memory resources that can be used by a VM. When multiple VMs are started, they will complete for memory resources. To improve memory utilization and reduce idle memory, users can set the memory limit parameters in the configuration file when creating a VM to ensure that the memory allocated to the VM does not exceed the upper limit.

Network QoS

The network QoS policy controls the bandwidth configuration. Network QoS is determined by the following aspects:

- Bandwidth control based on the sending or receiving direction of port group members
- Traffic shaping and bandwidth priority configured for each member port in a port group

6 Cloud Computing Trends

Cloud computing has laid a sound foundation for artificial intelligence (AI), and big data has fertilized the AI technology. To better support AI and big data, cloud computing needs to evolve to keep up with the rapid pace of technological advancements. This chapter describes the fields and key enabling technologies related to cloud computing and also introduces the technologies emerging from cloud computing.

6.1 Fields Related to Cloud Computing

IoT

As an important technology of today and the future, the Internet of Things (IoT) creates a network of things (or objects) that are connected over the Internet. This implies that: First, the Internet serves as the core and basis of IoT, while IoT is an extension of the Internet into physical devices and everyday objects. Second, IoT enables information exchanges and communication between objects, such as radio frequency identification (RFID), infrared sensors, global positioning systems, and laser scanners, based on agreed protocols. People can recognize, locate, track, monitor, and manage objects in IoT more intelligently.

Key technologies used by IoT:

- RFID: electronic tag, a type of smart card. RFID is used to enable IoT. The term IoT was coined by Kevin Ashton, Executive Director of MIT's Auto-ID Center, in 1999.
- Sensor technology: Various types of sensors are used to detect and collect environmental signals, including temperature, humidity, pressure, and speed.
- Embedded system: An embedded system is a dedicated computer system in which hardware and software are tightly integrated to offer specific functions. Each embedded system is just a part of a larger system.

Cloud Computing, Artificial Intelligence (AI), and Big Data

AI has always been hailed as a technology that can change the world or even end human beings. AI, in combination with cloud computing and big data, are driving a digital transformation of the ICT industry for higher intelligence.

Big data was originally an application of cloud computing and could not work without cloud computing. Big data was placed in the Technology Trigger phase of Gartner's 2011 hype curve for emerging technologies. In 2013, when cloud computing was in the Trough of

Disillusionment phase, big data just entered the Peak of Inflated Expectations phase. In 2014, big data entered the Trough of Disillusionment phase and began to grow at the same speed as cloud computing. Today, while cloud computing is still indispensable and fundamental to big data, big data is also enabling the success of cloud computing because it is an important type of workload on clouds.

The data volume was small at early stages of IT. People used to read books and newspapers. There were not even many books in a school's library. Now everyone reads e-books and news online. The amount of information increases with the advent of digitization.

Data in the realm of big data is categorized into the following:

- Structured data

  Data in a fixed format and with a limited length. For example, information in a table is structured data, such as values of nationality, ethnicity, and gender.
- Unstructured data

  Data with variable length and no fixed format, such as a voice message, a video, and a web page. For example, web pages are sometimes very long, and sometimes very short. Nowadays, the amount of unstructured data is increasing.
- Semi-structured data

  Data with certain structure in formats like XML or HTML.

In fact, some data is not useful until it is processed. Data is in the wristband if you wear one every day, and in web pages accessible over the Internet. Data itself is not useful, but it contains a very important thing called information.

Data is in disorder and can be called information only after being sorted and cleansed. Information contains many rules, and rules summarized from information are called knowledge. Knowledge changes how we live our lives, often for the better. While some people see information but don't know how to react, others see the future of e-commerce and live broadcasting. This is why the second type of people succeed. If you cannot extract knowledge from information, you are only a bystander in the Internet tide even though you browse through the social media every day.

We can apply knowledge to practice. Some people can do it very well, and that is intelligence. Knowledge does not necessarily mean intelligence. Many scholars are very knowledgeable and can analyze things from various perspectives. However, they cannot apply their knowledge to practice or transform it into intelligence. Many great entrepreneurs can do that and they finally succeed in business.

Therefore, there are four steps of data utilization: data, information, knowledge, and intelligence.

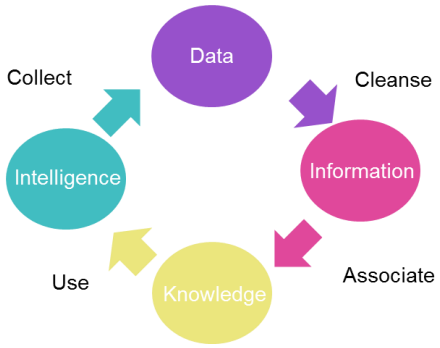


Figure 6-1 Four steps of data utilization

Many businesses want intelligence. They collect a huge amount of data and want to make business decisions and improve their products based on analysis of the data. For example, when a user watches a video, they can display an advertisement next to the video, which advertises exactly what the user wants to buy. For another example, when a user listens to a song, they can provide recommendations of similar music for the user.

Some companies extract information from the data users have entered on app interfaces or websites to predict user behavior and increase user stickiness. For example, an online shopping platform can make smart recommendations to a consumer based on his or her past behavior.

Many people think this is amazing, and wonder how it works.

The process includes data collection, transmission, storage, processing and analysis, retrieval and mining.

When the data volume is small, a few servers can do the job. As the data volume is increasing, more servers are needed.

- Data collection

  IoT needs thousands of sensing devices deployed to collect a large amount of temperature, humidity, monitoring, and power data. Search engines use multiple servers to form a web crawler system. The servers in this system work together to download all web pages in a limited timeframe.

- Data transmission

  Queues in memory are easily filled with a large amount of data. Therefore, distributed queues based on hard disks are created. In this way, data can be transmitted in queues of multiple servers at the same time. Massive data sets can be transmitted, as long as the queues and bandwidth are sufficient.

- Data storage:

  The file system of a single computer is rarely enough. Instead, a large distributed architecture is required to integrate hard disks of multiple computers into a large file system.

- Data analysis:

  One server cannot analyze, calculate, and summarize large data sets even over a very long period of time. Therefore, a distributed computing mechanism is invented to divide a large data set into smaller pieces. Each server processes a small piece of the data, and multiple servers process the data in parallel. In this way, the processing can be completed quickly. For example, TeraSort can sort 1 TB data, which is equivalent to 1000 GB, in 209 seconds using a distributed architecture where the data is concurrently processed by multiple servers. If the data is processed by one server, it will take several hours.

Now, let's talk about cloud computing.

A company may use big data to analyze its financial status once a week. It is wasteful to use many servers only once a week. Is it possible that we can allocate many servers for this weekly assignment and release them for other uses once this assignment is done? The answer is yes.

Cloud computing is the only way to do this. Cloud computing provides flexible, rapid elasticity needed for big data computations. Alternatively, big data services can also be deployed on a PaaS platform for on-demand access. In fact, big data services are important and widely used PaaS layer applications.

Big data is very complex and needs to be developed by professionals. Currently, big data solutions are available on public clouds. When a small company needs a big data platform, the

company does not need to purchase a lot of servers. Instead, they only need to access a public cloud provisioning the required big data services and have their data computed on that public cloud.

People can search for many things using Internet search engines powered by big data, but there are also many things that cannot be searched. Sometimes, people do not know how to search for the things they need, or things they find are not what they want.

For example, a music app can recommend a song that the user has never heard of and does not know its name but the user does like it. This is what search engines cannot do. When using such an app, people find that the app knows what they want before they do. They find that the app knows them like a friend. This recommendation function is only one of the many preliminary applications of AI.

People have been trying to develop AI for a long time. At first, people imagined a robot behind a wall and the robot would respond if people spoke to it. If people could not tell whether it is a human or machine from the responses, it is AI.

People come up with a way to program computers with the ability of human reasoning. Reasoning is the most important ability to human and also the difference between human beings and animals. If robots are enabled to reason and answer questions, it will be a great achievement.

In fact, machines have gradually acquired the ability to do some reasoning, such as proving mathematical formulas. It is a surprise that machines can prove mathematical formulas. But it turns out that the result is not so unexpected. People find that mathematical formulas are precise, and so is the reasoning process. It is easy for a machine to express mathematical formulas and programs.

But human language is not that easy. It is arbitrary and implied, making it difficult for machines to understand these meanings, but easy for human.

Therefore, for machines, reasoning without necessary knowledge is not enough. But it is a formidable challenge for ordinary people to impart such knowledge to machines. Probably, experts in linguistics or economics are able to do so.

Can linguistics and economics be more accurate, like mathematical formulas? If so, linguists can summarize the grammatical rules of the subject, predicate, object, attributive, adverbial modifier, and complement, and then computers can attempt to express according to these rules.

Later, people find it not feasible because it is too difficult to summarize linguistic rules as the meaning varies easily. For example, in spoken language, the predicate is sometimes omitted. In the meantime, you cannot ask users to speak standard written language to machines just for speech and semantic recognition purposes. This is not smart. As Luo Yonghao (a Chinese entrepreneur and Internet celebrity) once said in a speech, it is embarrassing to speak to his phone in a written-language style, like "Please call XX for me."

AI in this phase is called expert system. The expert system is not easy to achieve. On one hand, it is difficult to summarize knowledge. On the other hand, it is difficult to teach the summarized knowledge to computers. The reason is simple: How can computers be programed to abide by rules that are confusing even to people?

So people think that since machines are entirely different from humans, machines can be enabled to learn by themselves.

What enables machines to learn? Machines have a strong statistical ability. With this ability, they are able to find patterns from a large amount of data.

Of course, a real statistics-based learning algorithm is much more complex than simple statistics.

Statistical learning understands simple correlations. For example, if two words always appear together, then they are relevant. However, it cannot express complex ones. Usually, formulas of the statistical method are complex. To simplify calculation, various independent assumptions are often made to reduce the difficulty of formula calculation. However, in real life, independent events are relatively fewer.

So from the machine world, humans began to reflect on how the human world works.

A human brain does not store a large number of rules or statistic data, but is controlled by neurons. Each neuron has an input from another neuron. When receiving the input, the neuron generates an output to stimulate other neurons. So a large number of neurons react to each other, and finally various outputs are formed.

AI is able to do many things, such as identifying spam emails as well as pornographic, violent texts and pictures. This has experienced three stages.

- The first stage depends on the keyword blacklist, whitelist, and filtering technologies. Keywords can be set to pornographic or violent words. As the network language increases and words vary continuously, it is difficult to update the keyword library promptly.
- The second stage is based on some new algorithms like Bayesian filtering. It is a probability-based algorithm.
- The third stage is based on big data and AI for more accurate user profiles, content understanding, and image understanding.

Most AI algorithms depend on massive data sets for training and verification. Such data needs to be accumulated in a specific field (such as e-commerce and email) over a long period of time. Without data, AI algorithms are useless. Therefore, AI programs are totally different from IaaS and PaaS services provided by cloud computing in the sense that AI programs cannot be used on the customer's premises where the necessary data is missing.

Cloud computing vendors have accumulated huge amounts of data. Ordinary users can have access to AI capabilities provided on cloud platforms through exposed APIs. In cloud computing, this type of services is called software as a service (SaaS).

Finally, here are the three cloud computing service models: IaaS, PaaS, and SaaS. Generally, cloud, big data, and AI technologies can all be found working together in a cloud computing system. A big data company that has accumulated massive data amounts use AI algorithms to provide services, while an AI company mostly definitely runs a big data platform.

Copyright © Huawei Technologies Co., Ltd.

Cloud Computing, IoT, and Big Data

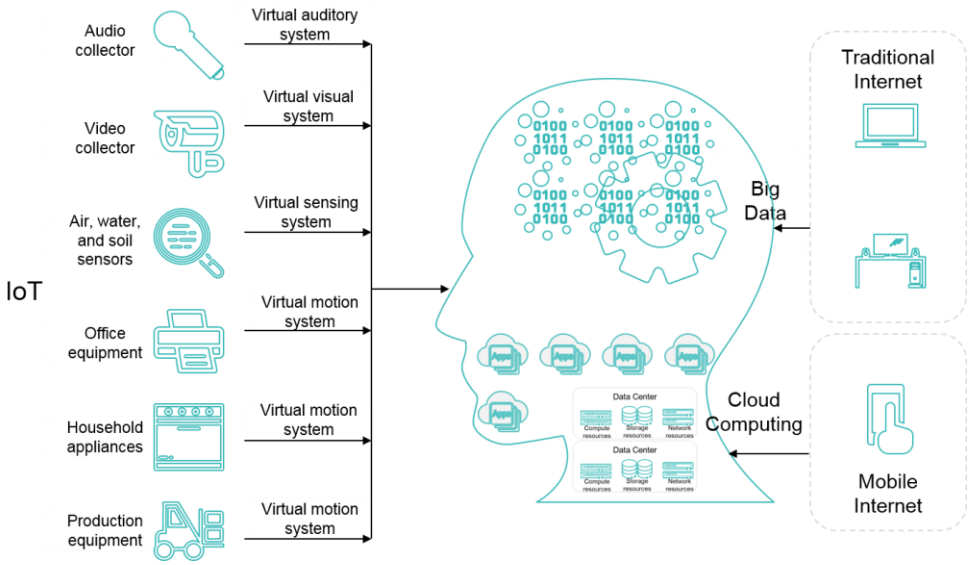


Figure 6-2 Relationship between cloud computing, IoT, and big data

As previously mentioned, cloud computing is the foundation of AI, big data fertilizes AI technology, while IoT provides the raw materials used to produce the fertilizer.

IoT is an important source of big data. IoT is the perception and motion detection system of the Internet. Big data is the information layer of the Internet and is the basis of Internet intelligence and consciousness. The rapid elasticity characteristic of cloud computing is required for data mining. Therefore, cloud computing and IoT promote the development of big data; in return, big data also accelerates the progress of cloud computing and IoT.

5G and Cloud Computing

Cloud computing is a product of the development of computing and the Internet, and the mobile Internet is ushering in the 5G era. "5G will empower information and communications technologies, and trigger many technological and commercial changes," said Ken Hu, Rotating Chairman of Huawei, at the Global MBB Forum 2018 in London.

5G will turn connectivity into a platform. With 5G, everything will go online and stay online by default. It is estimated that by 2025 there will be 40 billion intelligent devices on 5G networks. Ken Hu believes that in the future, it will be difficult to have no access to the Internet.

Ken Hu points out five fundamental changes that 5G will bring: connectivity as a platform, always-online, all-cloud, redefinition of terminal devices, and seamless experience. He also believes the world is embracing cloud technologies. This will allow everyone to access available information anytime and anywhere. New business models like Cloud X will emerge. In such models, all devices are cloud-based. Soon, there will be more Cloud X applications, such as Cloud PC Workstation, cloud gaming, and cloud VR/AR. Companies like Microsoft, Ubisoft, and EA are already working on these.

Copyright © Huawei Technologies Co., Ltd.

6.2 Cloud-Enabling Technologies

6.2.1 Container

A container is a lightweight OS-layer virtualization technology. It allows user space in an OS to be divided into several independent units running in the kernel, each of which is independent from each other. Such independent space is called a container.

Containers ensure that applications run in the same way everywhere. Therefore, developers can test containers on their own computers and run them without any modification on virtual machines (VMs) or other physical machines.

Containers are also a virtualization technology. Many industry discussions inevitably compare containers with VMs.

A container consists of:

- Application
- Runtime environment, including libraries

A container is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries, and settings. Available for both Linux and Windows-based applications, containerized software will always run the same, regardless of the infrastructure. Containers isolate software from its runtime environment and ensure that it works uniformly despite differences for instance between development and staging, minimizing conflict between applications on the same infrastructure. The primary difference between containers and VMs is that each VM includes the operating system (OS) but a container does not.

What benefits does the container technology provide?

Software architecture is so complex that environment configuration becomes an obstacle in developing software. Software can properly run in an environment only when the environment configuration, including OS settings, various libraries, and components, is correct. For example, to run Java-based software, the computer must have a matching engine, dependencies, and environment variables. To reduce environment dependency, someone advises to copy the original environment settings to the target computer where the software is to be installed.

And someone suggests that a VM template can also address this issue.

As we have mentioned, each VM includes an OS. Compared with containers, VMs have the following limitations:

1. High resource overheads

   The VM itself exclusively occupies part of the memory and disk space. When it runs, these resources cannot be used by applications. Even if the applications only need 1 MB of memory, the VM still uses hundreds of MBs of memory to run.

2. Redundant operations

   Each VM has an independent OS and some of the OS operations are inevitable, such as user login.

3. Slow startup

   It takes the same time to start an OS and VM, which may be a few minutes. The application can start to run only after the OS starts.

The limitations of VMs are fully addressed by containers. The benefits are as follows:

1. Quick startup

   Containerized applications are a process of the underlying system, not a process of the VM. Starting a container takes approximately the same time as starting a process, which is much faster than starting a VM.

2. Small size

   The container file size is much smaller than the VM file size since a container contains only the required components while a VM includes an entire OS.

3. Small resource overheads

   A container occupies only necessary resources. A VM has a complete OS and therefore occupies all resources needed to run an OS. In addition, containers can share resources, but each VM requires exclusive access to resources.

What is the relationship between Docker and containers?

Docker is an encapsulation of Linux container. It is currently the most popular Linux container solution and provides standard, easy-to-use APIs. Docker packages applications and their dependencies in a single file. When users run this file, a virtual container is generated. Applications run in the virtual container, just as they would on a real physical machine. Docker helps developers avoid the inconsistency of development environments. Generally, Docker APIs are simple. Users can easily create and use containers and put their applications in the containers. Containers can be copied, shared, and modified, just like code, and their versions can be managed.

Docker is mainly used for three purposes:

- To provide a one-time environment, for example, the environment for local software testing and continuous integration.
- To provide elastic cloud services. Docker containers can be enabled and disabled as required, suitable for dynamic scaling.
- To build the microservice architecture. With multiple containers, a single machine can run multiple services, forming a microservice architecture.

The core components of Docker include:

- Docker client
- Docker daemon
- Docker image
- Registry
- Docker container

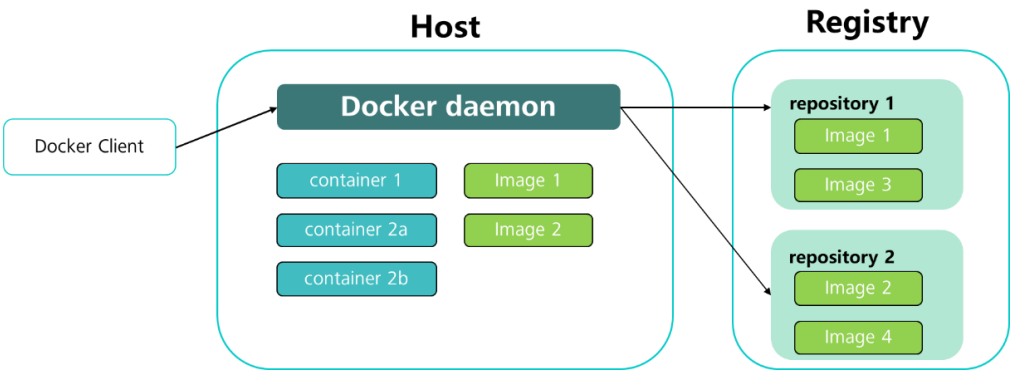


Figure 6-3 Docker architecture

Docker uses a client-server architecture. The client sends requests to the server. The server builds, runs, and distributes containers. The client can run on the same host as the server or connects to a remote server over sockets or using REST APIs.

- Docker client

  Users primarily use command lines to interact with the Docker server. The client sends commands to the server to build and run containers on the host.

- Docker daemon

  The Docker server component is the Docker daemon, which runs on the Linux operating system and is responsible for creating, running, and monitoring containers, as well as building and storing images.

- Docker image

  It is a key component of Docker and is used to create Docker containers. Docker images are read-only templates and similar to VM templates. To modify an image, users need to convert it to a container first. After the modification, convert it back to an image again.

- Docker container

  A Docker container is a runnable instance of an image.

- Registry

  A Docker registry stores Docker images. There are two types of registries: public and private.

6.2.2 OpenStack

6.2.2.1 What Is OpenStack?

OpenStack is the most popular open-source cloud operating system at present. OpenStack was first launched in June 2010 and has since become a mature platform. OpenStack provides a variety of powerful functions and has been widely used to build private clouds, public clouds, and NFV architectures. In addition, OpenStack has the support of almost all mainstream vendors in the IT industry, and spawned many startups providing related products and apps. In fact, OpenStack has become the mainstream standard in the open-source cloud computing industry. Today, a prosperous and sustainable OpenStack ecosystem has formed. OpenStack has become an inevitable topic in the cloud computing era. More to the point, OpenStack is principal for understanding the development and trends of cloud technologies. This chapter describes a few key points concerning OpenStack.

OpenStack is officially defined as a cloud operating system on https://www.openstack.org/. The following uses a PC OS as an example to explain cloud OS.

An OS is crucial for a computer system. Without an OS, the hardware and software of a computer cannot be integrated into a system to handle tasks and serve users. OSs commonly used include Linux and Windows running on servers and PCs, as well as Android and iOS on mobile phones. An OS provides the following functions: resource access and abstraction, resource allocation and scheduling, application lifecycle management, system management and maintenance, and man-machine interaction. Each of the functions is indispensable.

To be more specific: (1) Resource access and abstraction: connects hardware devices, such as CPUs, memories, local hard disks, and network adapters, and abstracts them as logical resources that can be managed by the OS. (2) Resource allocation and scheduling: allocates the hardware resources to software or applications by using OS resource management capabilities based on the types and quantities of resources needed. (3) Application lifecycle management: helps users install, upgrade, start, stop, and uninstall applications on the OS. (4) System management and maintenance: helps system administrators configure, monitor, and

upgrade the system. (5) Man-machine interaction: provides man-machine interfaces for system administrators and users to perform necessary operations.

A cloud OS has similar functions. The primary difference is that a cloud OS manages a distributed cloud computing system composed of a large quantity of software and hardware, whereas a common OS manages a local server, PC, or mobile phone.

For a cloud OS, its main functions are as follows: (1) Resource access and abstraction: connects virtualized or software-defined hardware resources, such as servers, storage devices, and network devices to a cloud computing system, and abstracts them into compute, storage, and network resource pools that are recognizable to the cloud OS. The cloud OS manages the resource pools to control the underlying hardware. (2) Resource allocation and scheduling: allocates compute, storage, and network resources to tenants and applications by using cloud OS resource management capabilities. (3) Application lifecycle management: helps tenants install, start, stop, and uninstall cloud applications in the cloud OS. (4) System management and maintenance: helps system administrators manage and maintain a cloud computing system. (5) Man-machine interaction: provides man-machine interfaces for system administrators and common tenants to perform necessary operations.

The cloud OS is far more complex than the conventional OS used in daily life, but they provide similar functions. OpenStack provides the framework for building a cloud OS.

To build a complete cloud OS, a large number of software components need to be integrated and work together to provide functionalities and services required by system administrators and tenants. However, OpenStack cannot independently provide all the capabilities required by a cloud OS. Specifically, OpenStack cannot independently carry out resource access and abstraction but must work with underlying virtualization software, software-defined storage, and software-defined networking (SDN). It cannot independently manage application lifecycles but needs to integrate management software platforms at the upper layer. It does not have comprehensive system management and maintenance capabilities and must integrate various management software and maintenance tools. OpenStack provides its own man-machine interfaces, but they are inadequate for commercial systems.

In short, to build a full-fledged cloud OS, OpenStack must be integrated with other software components. Therefore, OpenStack is positioned as the framework for cloud OS. Different components can be integrated on this framework to meet customer needs for cloud OSs.

Open source lies in the core of OpenStack. It is also the key to understanding the past and future of the OpenStack community. Different from the source code extensively released on the Internet, the OpenStack community follows a more profound open source concept. In the OpenStack community, the entire process of code development, from requirement submission, scenario analysis, solution design, code submission, test execution, to code integration for each component and feature, as well as each line of code, complies with the community's openness principle and is visible to the public, allowing maximum transparency and participation from contributors. This has kept the community open and prevented the community from being controlled by a small group of people, companies, or organizations, ensuring the sustainable growth of the community ecosystem. In addition, all of the code for OpenStack is under Apache 2.0 license. This guarantees commercial benefits for enterprises participating in the community and promotes commercial success of OpenStack products. OpenStack is the framework software that is developed and released as open source and designated to build cloud OSs in different scenarios. An in-depth understanding of this concept is very important for further studying OpenStack.

Copyright © Huawei Technologies Co., Ltd.

6.2.2.2 Relationship Between OpenStack and Cloud Computing System

OpenStack is closely related to but still different from a cloud computing system.

OpenStack is a framework for building cloud OSs. To set up a cloud computing system, a cloud OS must be integrated with hardware devices and run various types of upper-layer applications and services on top.

6.2.2.3 Relationship Between OpenStack and Compute Virtualization

Compute virtualization is a concept that many readers are familiar with. The corresponding software is the Hypervisor, such as the open-source KVM and Xen, VMware vSphere, Huawei FusionCompute, and Microsoft Hyper-V. The relationship between OpenStack and compute virtualization is still confusing to many people. The following explains their relationship to help readers better understand OpenStack. OpenStack is a framework for building cloud OSs. To build a complete cloud OS, especially to implement resource access and abstraction, OpenStack must integrate with virtualization software to achieve compute resource pooling of servers. The virtualization software carries out the task of virtualizing physical resources and putting them into respective pools. If KVM is used as the virtualization software of OpenStack, KVM virtualizes a physical server into multiple VMs while OpenStack maintains the resource pools, for example, recording the quantity of servers in the system, the amount of resources on each server, amount of allocated resources, and amount of idle resources. OpenStack delivers control commands to KVM and performs VM lifecycle management operations, such as creating, deleting, starting, and stopping VMs. This shows that OpenStack is like the brain of a cloud OS and compute virtualization software is more like the body that carries out specific operations. To properly manage compute resources in a cloud computing system, OpenStack must work with compute virtualization software. However, the two are not the same.

6.2.2.4 OpenStack Design

OpenStack has been developing rapidly, attributed to the blooming of cloud computing technologies, as well as its unique design. OpenStack has an open, flexible, and scalable design.

1. Open

   OpenStack is open to the public, including its source code and process of design, development, testing, and release. This avoids OpenStack from being controlled by individuals/enterprises or evolving towards a closed architecture/system. OpenStack provides open northbound APIs allowing access of southbound hardware and software. To make design and development more efficient and improve software quality, OpenStack adheres to the principle of "Not Reinventing the Wheel" and continuously introduces and reuses excellent open-source software from various technical fields.

2. Flexible

   Many of the OpenStack components are pluggable and configurable. OpenStack uses plugins to manage and connect diverse compute, storage, and network resources and uses a single architecture to pool heterogeneous resources from different vendors. For compute resources, OpenStack can integrate hypervisors, such as KVM, Xen, vCenter, and FusionCompute as plugins. For storage resources, it can manage storage devices of different vendors and software-defined storage like Ceph, FusionStorage, and vSAN through plugins. For networking resources, it allows access of hardware network devices, open-source network components like OVS, Linux bridge, and HAProxy, as well as SDN controllers. The accesses are all configurable. To connect to new resources, developers modify configuration items to choose appropriate plugins for OpenStack, without the need to repackage it.

In addition, OpenStack does not depend on any commercial software or hardware. In other words, any commercial software and hardware products are replaceable in the OpenStack architecture. Users can use open-source solutions to build OpenStack-based cloud computing systems without worrying about vendor lock-in.

3. Scalable

   OpenStack is highly scalable in terms of functionalities and capacities. From the perspective of functionalities, OpenStack consists of multiple decoupled projects. Each project implements unique functions in a cloud computing system, such as the identity authentication and authorization service, compute service, block storage service, network service, image service, and object storage service. For a cloud computing system used in a specific scenario, system designers can deploy OpenStack projects as needed, or introduce new projects after the system is released. Certain OpenStack projects also provide scalable functions. System designers can introduce new function modules to these projects and extend their functions without affecting the existing functions. From the perspective of scalable capacities, OpenStack adheres to the principle of designing a centerless and stateless architecture. Most OpenStack projects can be horizontally scaled to build cloud computing systems of different sizes. After deployment, a cloud computing system can be scaled on demand by adding both management and resource nodes. Using this architecture can effectively cut upfront investment, reduce the complexity of capacity planning, while providing sufficient room for future scalability.

6.2.2.5 OpenStack Architecture and Projects

When the OpenStack community launched the first release Austin in 2010, OpenStack included only two projects Nova and Swift, only providing simple and basic functions. Now, OpenStack is mature enough to support an increasing number of projects. OpenStack Mitaka released up to 29 service projects. Each service project has distinct functions and they work together to build a cloud OS with a flexible, multifunctional, and highly scalable architecture.

To provide a brief overview of OpenStack, this section describes its most important and typical service projects.

1. Keystone: identity, authentication, and access management service

   A cloud OS has fundamental capabilities of sharing compute, storage, and network resources as well as IaaS, PaaS, and SaaS built on the resources among users and enabling users to securely access and use the same cloud computing system. To realize the capabilities, Keystone was released, functioning as an identity, authentication, and access management service project. It authenticates user identities and issues tokens to authorized users. Users use the tokens to access other OpenStack projects. Together with token authentication and permission control mechanisms embedded in each component, Keystone authenticates user identities and controls their permissions. Each authorized user is given the ability to perform operations allowed by their permissions for designated resources, isolating and protecting user resources.

2. Nova: compute service

   Nova is the OpenStack project that provides a way to provision compute instances, allowing users to create VMs as needed. Nova manages a large number of physical servers deployed with compute virtualization software (hypervisors) and integrates underlying resources into a logical resource pool with a resource view. Based on this, Nova manages the lifecycle of resources in the resource pool to fulfill different user requests. The primary management operations include creating, deleting, starting, and stopping VMs. When receiving a VM creation request, Nova integrates the CPU, memory, local storage, and I/O devices in the logical resource pool to create VMs of different flavors, installs OSs on the VMs, and finally provides a VM satisfying the user requirements for compute resources.

In addition to VM resource management, Nova works with Ironic to provide bare-metal resource management services for users. Specifically, when Nova receives user requests for bare-metal resources, it can invoke Ironic functions to implement automated bare-metal selection, allocation, and OS installation, offering the same experience in using virtual machine resources and physical machine resources.

3. Ironic: bare metal

   Ironic works with Nova to provide the bare metal service for users.

   In practical use, Ironic manages physical servers. When a physical server is added to a resource pool, Ironic records the hardware specifications of the physical server and reports the specifications to Nova. When a user initiates a bare metal management operation, Ironic executes operations for the physical server according to the command from Nova. For example, when an action for creating a bare metal server is triggered, Ironic performs operations, such as initializing hardware configuration and installing the OS, for the selected physical server according to the Nova command.

4. Glance: image

   Generally, each VM needs to be installed with an OS after being created. To suit diverse user demands, a cloud computing system is preconfigured with multiple OS images running different versions and the images are often installed with common software. To manage various images, the OpenStack image service, Glance, is introduced.

   Glance manages metadata of created images and is capable of creating, deleting, querying, uploading, and downloading images. In a production environment, Glance does not store image files, but is only responsible for storing metadata of image files. In essence, Glance is a management frontend. To provide image management and storage service capabilities, Glance needs to be connected to object storage backends.

5. Swift: object storage

   Object storage is a common data storage service in cloud computing. It is deployed in scenarios where a single file with a large amount of data needs to be stored, where data is less frequently accessed, where requirements for data access latency are low, and where data needs to be stored with lower costs. Swift is an OpenStack project that provides the object storage service.

   Unlike most OpenStack projects that implement only control functions and do not directly carry user services, Swift provides a complete object storage system and can be independently deployed.

   In addition, Swift can be used as the backend storage of Glance to store image files.

6. Cinder: block storage

   In OpenStack projects using KVM, VMs created by using Nova use the local file system of each compute node as its data storage by default. The lifecycle of the data store is the same as that of the VM. In other words, the data store is removed once the VM is removed. If users require persistent block storage media that are independent of VM lifecycle, Cinder can be used to provide the block storage service (also called volume service).

   Cinder defines storage capabilities provided by backend storage devices or software-defined storage clusters as block storage resource pools. Then, based on user requirements, it divides resource pools into volumes with different sizes and allocates them to users.

   When using volumes provided by Cinder, users need to attach the volumes to VMs by means of Nova. In the VM OSs, they can view and access the block devices corresponding to the volumes.

7. Neutron: networking

Networking is crucial to IaaS capabilities of all cloud OSs. Cloud computing systems must be built on stable, easy-to-use, and high-performance virtual networks to provision resources and services for users.

Neutron is an OpenStack project that provides the network service. Neutron and its sub-projects provide users with network service capabilities from Layer 2 to Layer 7, including Layer 2 networking, Layer 3 networking, internal DHCP management, Internet floating IP address management, internal and external firewalls, load balancing, and VPN. Layer 2 and Layer 3 service capabilities of Neutron are mature enough to replace NovaNetwork and become the mainstream Layer 2 and Layer 3 virtual network service in OpenStack. Its service capabilities from Layer 4 to Layer 7 are being developing rapidly and now can be preliminarily put into commercial use.

It should be noted that DNS as a service is not included in Neutron, but is provided by an independent project Designate.

8. Heat: orchestration

   One of the core values of cloud computing is automated provisioning and management of IT resources and services. After cloud computing was introduced, a large number of complex manual tasks in traditional IT can be automatically completed by calling APIs provided by cloud OSs to improve the IT system management efficiency.

   Among these tasks, lifecycle management, including installation, configuration, capacity expansion, and removal of application systems, is typically complex and time-consuming, which cannot support rapid service provisioning or elastic scaling. OpenStack Heat has come into being to provide automated lifecycle management capabilities for application systems. Specifically, Heat parses a submitted template that define resource types, quantities, and connections required by an application system, and calls Nova, Cinder, and Neutron APIs to enable automated deployment of the application system. This process is highly automated and programmed. The same template can be reused in any OpenStack-based cloud computing systems, remarkably improving the deployment efficiency of application systems. Heat can also work with the Aodh subproject of OpenStack Ceilometer to enable autoscaling for application systems, facilitating the management of stateless and horizontally scalable application systems.

9. Ceilometer: metering and monitoring

   In cloud computing systems, resources are provisioned as services to users, and users need to pay fees based on the resource types and quantities they need. For this purpose, a cloud OS must be able to monitor and measure resource usage. This is why OpenStack has introduced Ceilometer.

   Ceilometer mainly collects information about the types and quantity of resources used by users in polling mode. The collected information is used as the basis for charging.

   Based on the collected information, Ceilometer sends alarm signals through the Aodh sub-project to trigger Heat to execute autoscaling.

   Ceilometer does not provide the billing capability. To support this function, system designers need to connect Ceilometer to a proper billing module. The OpenStack community has created the project CloudKitty as a native billing component. However, the project is still in the early stage and cannot be put into commercial use.

10. Horizon: dashboard

    Horizon provides a web-based graphical user interface to OpenStack services. Horizon provides administrators and users with a simple and user-friendly interface and serves as the basic portal for managing OpenStack-based cloud computing systems.

    In addition, Horizon has a pluggable architecture, making it easy for designers to incrementally develop the system based on user requirements.

NOTE

Section 6.2.2 is excerpted from *Cloud Computing Architecture Technologies and Practice.*

6.3 Other Emerging Technologies

6.3.1 Fog Computing

Fog computing is an extension of the concept of cloud computing and was proposed by Cisco. In fog computing, data and application sets are stored in edge devices, not clouds. The term "fog" was chosen because fog is a cloud closer to the ground.

Fog computing was coined by Stolfo, Professor of Columbia University in New York, for the purpose to protect against hacking attacks. Later Cisco officially proposed and redefined the concept of fog computing. Fog computing uses a decentralized, IoT-oriented infrastructure extending computing capabilities and data analytics applications to the network edge. Fog computing enables local data analytics and management and delivers real-time insights on data to customers.

The name fog computing is as vivid as cloud computing. As the names imply, the cloud is located up in the sky, somewhere distant and remote, whereas the fog is close to the ground where servers are located near end users. Fog computing utilizes small, geographically-dispersed computers penetrating into various industries like factories, automobiles, electrical appliances, street lights, and commodities.

6.3.2 Edge Computing

Edge computing is one of the future trends of cloud computing, but it is still in a conceptual phase.

According to Wikipedia, edge computing is defined as a distributed computing paradigm which brings compute and storage resources closer to where they are needed.

Edge computing, to some extent, is decentralized or distributed cloud computing. In edge computing, raw data is analyzed locally instead of being transferred to the cloud. Edge computing advocates believe that computing power is moving from the cloud to the edge, and that edge computing will be the next thing after cloud computing. An important rationale behind this is that edge computing plays a critical role in driving IoT development.

Edge is a general concept. It refers to the computing infrastructure that is close to data sources. Different edge computing providers have different definitions of what edge is. For example, for the US telecom company AT&T, the edge refers to cellular base stations deployed a few miles away from the customer. For Akamai, the largest CDN vendor in the world, the edge refers to CDN devices distributed all over the world. For monitoring equipment at airports, the edge refers to HD cameras providing full coverage with no blind spots.

From the viewpoint of *Fog Computing vs. Edge Computing: What's the Difference?* (https://www.automationworld.com/fog-computing-vs-edge-computing-whats-difference), "both fog computing and edge computing involve pushing intelligence and processing capabilities down closer to where the data originates, many industries use the terms fog computing and edge computing (or edge processing) interchangeably, and the key difference between the two architectures is exactly where that intelligence and computing power is placed."

- Fog computing brings compute capabilities to the local area network (LAN) and carries out data processing on fog nodes or IoT gateways.

- Edge computing pushes the intelligence, processing capabilities, and communication capabilities of edge gateways directly into devices like programmable automation controllers (PACs).

6.3.3 Microservices

Microservices are an architecture in which a monolithic application is divided into multiple small services. These services coordinate and cooperate with each other to provide users with the required functionalities. Each service runs in its own independent process and communicates with each other through a lightweight mechanism (usually HTTP-based RESTful API). Each service is built around a specific functionality and can be deployed independently to a production environment, pre-production environment, or another type of environment. In addition, unified and centralized service management mechanisms should be avoided as far as possible. For a specific service, appropriate language and tools should be selected to build it according to the business context. As one of the most popular buzzwords in the field of software architecture, microservices implement cloud-based software and focus on dividing complex applications into small-granularity, lightweight, and autonomous services.

In practice, microservices have the following characteristics:

1. Small

   By means of service-oriented analysis and service modeling, microservice divides complex business logic into a group of small, designated, loosely coupled, and highly autonomous services. Each service is a small but integrated application, which makes microservices different from components, plugins, and shared libraries. There is no formal definition of the microservice scale. The upper limit of the microservice scale is represented by the time a development team spends on restructuring a microservice. For example, a microservice can be restructured within two weeks by a microservice development team consisting of six to eight engineers.

2. Independent

   Microservices can be developed, tested, and deployed independently of each other. Each service in the microservice architecture is an independent unit, which is deployed as an independent service process. When the code of a microservice is modified, there is no impact on other microservices, because each microservice has an independent codebase. An independent test and verification mechanism is provided for each microservice, avoiding the need to perform a large-scale regression test for an entire system. (An integration regression test covering all system functions is rather time-consuming and probably generates inaccurate results.)

3. Lightweight

   Service autonomy is important for microservices. A lightweight communication mechanism (usually RESTful API) is leveraged for microservices to communicate with each other efficiently through messages. Such communication mechanisms are language- and platform-independent, which makes it convenient for developers to design communication protocols and maintain forward compatibility of interfaces.

   During the evolution from traditional software architecture to microservice architecture, industry practices maintain Remote Procedure Call (RPC) and formulate communication protocols based on RPC to ensure forward compatibility of interfaces. In other words, RPC-based communication protocols allow for independence and loose coupling between services.

4. Loosely coupled

   Microservices are loosely coupled. Each microservice can be independently deployed in any sequence. APIs between microservices are forward compatible with future versions. Each microservice supports independent gray release and gray upgrade.

To allow for loose coupling, each microservice should be designed for a single purpose. Independence of service logic is the key to decoupling microservices.

6.3.4 Serverless

Serverless architecture and Functions-as-a-Service (FaaS) are two upcoming cloud computing models. The following introduction to serverless computing and FaaS comes from Fei Lianghong, a Technical Evangelist at Amazon Web Services.

Like many new concepts, serverless architecture does not yet have a universally accepted definition. The latest definition is as follows: "Serverless architectures are Internet based systems where the application development does not use the usual server process. Instead they rely solely on a combination of third-party services, client-side logic, and service hosted remote procedure calls."

In the beginning, serverless architectures are designed to free developers from configuring and managing servers need to run backend applications. It does not aim to achieve "serverless" in the real sense. Instead, it means that a third-party cloud computing provider is responsible for maintaining backend infrastructure and providing developers with functions as services, such as database, messaging, and identity authentication. To sum up, this architecture is designed to allow programmers focusing on code running without the complexity of building and maintaining the infrastructure typically associated with developing and launching an application. Program code is deployed on a platform such as AWS Lambda, and functions are triggered by events. This architecture supports event-driven function calling and allows programmers to set timeout values for functions. For example, the default runtime of an AWS Lambda function ranges from 3 seconds to 5 minutes. Over the past two years, this architecture has been widely used, for example, to build a backend for mobile or IoT applications. Serverless architecture is not intended to replace traditional applications. However, developers and architects should pay attention to this new computing model as it has the potential to help improve efficiency and reduce costs and complexity.

Microservices allow developers to break applications down to their smallest components that are independent from each other, with each component focuses on a single functionality. These microservices are then assembled into complex, large-scale applications in a modular mode. On top of microservices, serverless further provides a software architecture featuring a high degree of "code fragmentation", called Function as a Service (FaaS). "Functions" are program units that are smaller than microservices. For example, you can use microservices to run the code required by CRUD operations (create, retrieve, update, and delete) or use "Functions" in FaaS to perform these operations. When the "create an account" event is triggered, a corresponding function is executed on AWS Lambda. Thus the serverless architecture and FaaS can be viewed as similar concepts.

Copyright © Huawei Technologies Co., Ltd.

7 Conclusion

Cloud computing has developed into what it is today thanks to a great number of technologies, not limited to those we have already discussed in previous chapters. This guide has focused on virtualization and technologies related to it. Container, OpenStack, and other proven technologies will be further discussed in our latest HCIP courses. For other emerging technologies, such as fog computing, edge computing, and Serverless, you can study them by reading documents and posts available on the Internet or Huawei official websites.

You can also join us at the official HCIA-Cloud Computing forum on the Huawei Enterprise Community website to communicate with peers. The address is as follows:

https://forum.huawei.com/enterprise/zh/thread-513763-1-1.html

If you have any suggestions, comments, or feedback about our courses or about cloud technologies, feel free to leave messages.

You can also obtain more information from the following websites:

Huawei Certification: https://support.huawei.com/learning/zh/newindex.html

Huawei e-Learning: https://ilearningx.huawei.com/portal/#/portal/EBG/26

Huawei ICT Academy: https://www.huaweiacad.com/webapps/hw-toboo-BBLEARN/portal/exec.do?t=1561101910908

You can also follow us at the following accounts:

Huawei Certification Facebook:

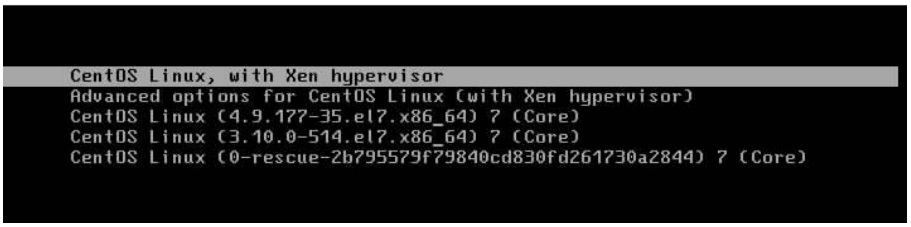
Huawei Certification Twitter:

Huawei ICT Academy Facebook:

8 Appendix

8.1 Verification 1

Xen and KVM have different architectures and yet both run in the Linux kernel. A Linux OS cannot run both hypervisors at the same time, so you have to choose one during startup.



That is to say, you can run KVM in a Linux OS as long as this OS does not run Xen. To determine the hypervisor used, run the **lscpu** command to check **Hypervisor vendor**. First let's see a sample of the command output when Xen is used (pay attention to the information in **bold**).

```
[root@localhost ~]# lscpu
Architecture:        x86_64
CPU op-mode(s):      32-bit, 64-bit
Byte Order:          Little Endian
CPU(s):              32
On-line CPU(s) list:  0-31
Thread(s) per core:   2
Core(s) per socket:   8
Socket(s):           2
NUMA node(s):        1
Vendor ID:           GenuineIntel
CPU family:          6
Model:               79
Model name:          Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz
Stepping:            1
CPU MHz:             2095.164
BogoMIPS:            4190.32
Hypervisor vendor:    Xen
Virtualization type:  none
L1d cache:           32K
L1i cache:           32K
L2 cache:            256K
L3 cache:            20480K
```

```
NUMA node0 CPU(s):     0-31
Flags:fpu de tsc msr pae mce cx8 apic sep mca cmov pat clflush acpi mmx fxsr sse sse2
ht syscall nx lm constant_tsc arch_perfmon rep_good nopl nonstop_tsc pni pclmulqdq est
ssse3 fma cx16 sse4_1 sse4_2 movbe popcnt aes xsave avx f16c rdrand hypervisor lahf_lm
abm 3dnowprefetch epb fsgsbase bmi1 hle avx2 bmi2 erms rtm rdseed adx xsaveopt cqm_llc
cqm_occup_llc cqm_mbm_total cqm_mbm_local dtherm arat pln pts
```

Run the same command on a CNA node and check the output:

```
CNA:~ # lscpu
Architecture:        x86_64
CPU op-mode(s):      32-bit, 64-bit
Byte Order:          Little Endian
CPU(s):              32
On-line CPU(s) list:  0-31
Thread(s) per core:   2
Core(s) per socket:   8
Socket(s):           2
NUMA node(s):        2
Vendor ID:           GenuineIntel
CPU family:          6
Model:               63
Model name:          Intel(R) Xeon(R) CPU E5-2640 v3 @ 2.60GHz
Stepping:            2
CPU MHz:             2600.609
BogoMIPS:            5193.14
Virtualization:      VT-x
L1d cache:           32K
L1i cache:           32K
L2 cache:            256K
L3 cache:            20480K
NUMA node0 CPU(s):    0-7,16-23
NUMA node1 CPU(s):    8-15,24-31
```

Run **lsmod** to check if the **KVM** module is loaded.

```
CNA:~ # lsmod | grep kvm
kvm_intel           176445  6
kvm                 598042  1 kvm_intel
irqbypass            13503  5 kvm,vfio_pci
```

The output indicates that the KVM virtualization architecture is used on CNA nodes in FusionCompute 6.3.

8.2 Verification 2

Run **qemu-kvm-version** on a CNA node to verify if the **qemu** module is used.

```
CNA:~ # qemu-kvm -version
2019-06-06T10:52:17.263814+08:00|info|qemu[15147]|[15147]|scan_dir[163]|: no
hotpatch direcToRy, will not reload hotpatch
2019-06-06T10:52:17.264967+08:00|info|qemu[15147]|[15147]|main[3342]|: qemu pid is
15147, options parsing start!
QEMU emulaToR version 2.8.1.1(25.165)
Copyright (c) 2003-2016 Fabrice Bellard and the QEMU Project developers
```

The output indicates the QEMU version is 2.8.1.1, indicating that the CNA node uses the **qemu** module and the KVM architecture.

In Linux OS, running **systemctl** displays all the services in use. Therefore, run this command on the CNA node to check if relevant services are running.

```
CNA:~ # systemctl | grep libvirt
libvirt-guests.service
loaded active exited   Suspend/Resume Running libvirt Guests
libvirtd.service
loaded active running  Virtualization daemon
```

The output indicates that libvirtd.service is in the running state, indicating that libvirt is also used on the CNA node.

8.3 Verification 3

Log in to a CNA node and run **ovs-vsctl show**.

```
CNA:/etc/libvirt/qemu # ovs-vsctl show
248b2f99-7edb-4ac4-a654-c9054def32bb
   Bridge "br1"
      Port "tap00000002.0"
         tag: 0
         Interface "tap00000002.0"
            options: {brd_ratelimit="0"}
      Port "patch-1"
         Interface "patch-1"
            type: patch
            options: {peer="patch-bond13"}
      Port "br1"
         Interface "br1"
            type: internal
      Port "tap00000001.0"
         tag: 0
         Interface "tap00000001.0"
            options: {brd_ratelimit="0"}
   Bridge "br-bond13"
      Port "Mgnt-0"
         Interface "Mgnt-0"
            type: internal
      Port "patch-bond13"
         Interface "patch-bond13"
            type: patch
            options: {peer="patch-1"}
      Port "br-bond13"
         Interface "br-bond13"
            type: internal
      Port "bond13"
         Interface "bond13"
ovs_version: "2.7.3"
```

A tap device usually indicates a VM NIC. Pay attention to the information in **bold**. Each NIC has two attributes: **tag** and **options**. **tag** refers to the VLAN tag set for the port, and **options** indicate QoS and type. You can add VM NICs to a port group when creating the VMs. This

way, you can set the port attributes simply by configuring this port group. Otherwise, you will need to configure attributes for each port individually.

8.4 Verification 4

Bridges do not support traffic statistics, so virtualization vendors collect traffic statistics from virtual switches. In FusionCompute, VRM collects port traffic statistics from distribute virtual switches (DVSs). Here, run **vs-ofctl dump-ports br1** to view the traffic information obtained by the DVS.

```
CNA:/etc/libvirt/qemu # ovs-ofctl dump-ports br1
OFPST PORT reply (xid=0x2): 4 ports
  port  2: rx pkts=2681611, bytes=676430572, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=3750074, bytes=1031065371, drop=0, errs=0, coll=0
  port  1: rx pkts=4521055, bytes=1094061951, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=2681620, bytes=676432674, drop=0, errs=0, coll=0
  port  3: rx pkts=9, bytes=2102, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=35, bytes=2202, drop=1053200, errs=0, coll=0
  port LOCAL: rx pkts=1437344, bytes=84784444, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=0, bytes=0, drop=0, errs=0, coll=0
```

Compare the output with the traffic information VRM displays, as shown in the figure below.

NIC	Port Group	DVS	Packets Received ...	Data Received (K...	Packets Sent (pac...	Data Sent (KB)	Outbound Packet...	Inbound Packets ...
Network Adapter 0	managePortgroup	ManagementDVS	3751825	1007390.72	2682880	660930.56		

The traffic statistics (such as throughput) of this NIC are close to those of port2 in the output above. Therefore, it can be concluded that the VRM node collects port traffic statistics from DVSs.