SANGFOR
深信服科技

SANGFOR

Hyper-Converged Infrastructure

White Paper

Sangfor Technologies Co., Ltd

Jan 7th, 2017

Copyright

Disclaimer

This document was prepared by Sangfor Technologies Co. Ltd. The information obtained herein is provided on an 'as available' basis. Sangfor Technologies Co. Ltd may make improvement or changes in this document, at any time or without notice.

The information is believed to be accurate. However, Sangfor shall not assume responsibility or held liable for any loss or damage resulting from omissions, inaccuracies or errors contained herein.

Contact Us

For any feedback or suggestion, please contact us through the following:

Address: iPark, A1 Building, Xueyuan Bvld 1001, Nanshan District, Shenzhen, Guangdong, PRC

Postcode: 518055

Tel: +86 755 26581949

Fax: +86 755 26581959

Website: www.sangfor.com

Abbreviation

Abbr.	**Full Name**
Hypervisor	Hypervisor
VMM	VMM Virtual Machine Manager
HA	High Availability
vMotion	vMotion
DRS	Distributed Resource Scheduler
RAID	Redundant Arrays of Independent Disks
IOPS	Input/Output Operations Per Second
VM	Virtual Machine
SDN	Software Defined Network
NFV	Network Function Virtualization

Revisions

Version	Drafted by	Date	Remarks
V1.1	Cheney Hu	2017-1	

Table of Contents

1. Preface

1.1 The IT Evolutions

Since the 1990s of the 20th century when Windows operating systems were widely used and Linux operating systems settled the leading position among x86 system, x86 system deployment has encountered new infrastructure and operating bottlenecks because of its rapid growth, including low infrastructure utilization, increasing hardware investment costs and IT operational costs, bad application failover and low disaster recovery capability, and so on.

As performance of x86 system is enhanced day by day, enterprises could eventually indulge into business investment rather than time saving and cost control. The major enhancements are, x86 system becomes generic and shared infrastructure, more hardware potentials are found, hardware utilization increases greatly, capital investment and operational costs are reduced dramatically, and system maintenance becomes simpler.

In addition, cloud computing and virtualization technology are leading data center development to a new era. Based on virtualization technology, management and business are centralized, and resources in data center are allocated and scheduled automatically, both of which cater enterprises' needs for high performance, reliability, security and adaptivity during crucial application migration to x86 platforms. Meanwhile, both trends are making infrastructure management more adaptive and automated to catch up with the fast business growth and increasing cloud solutions.

Cloud computing is not a brand new technology, but a solution coming into existence due to new drivers.

Traditionally, in order to provision new services, enterprises have to

start from networking plan and scale, choosing hardware models, making an order, paying their hardware supplier, making shipment, installing software, deploying hardware, and end by doing debugging.

The whole purchase and deployment cycle often takes up to weeks or months, for a large-scale project, which, however, could be shortened to minutes with the help of cloud computing.

> Moore's Law describes that chip performance would double every 18 months. Reverse Moore's law observes that as processors become faster and memory becomes cheaper, software becomes correspondingly slower and more bloated, using up all available resources. If software vendors become correspondingly slower and unable to catch up with the pace, they will be left behind in IT industry. IT industry is full of fierce competitions. Making use of cloud computing can upgrade efficiency of the IT infrastructure, but slows down product improvement or service development without.

Nowadays, we are amidst an enterprise-grade data center evolution that is only seen in decades, driven by the breakthrough of the 'software-defined' infrastructure. Computing capacity, networking and storage could be virtualized, allocated, and re-allocated, without any compromise with static hardware infrastructure. Software-Defined Data Center (HCI) enables enterprises focus on application, while IT resource scheduling is accomplished accordingly by the software dynamically.

Sangfor , the HCI Platform is a mature HCI solution. In addition to the common functionality, virtualization, standardization, automation, it boasts another four characteristics, simplicity, usability, security and reliability.

1.2 Sangfor HCI White Paper at a Glance

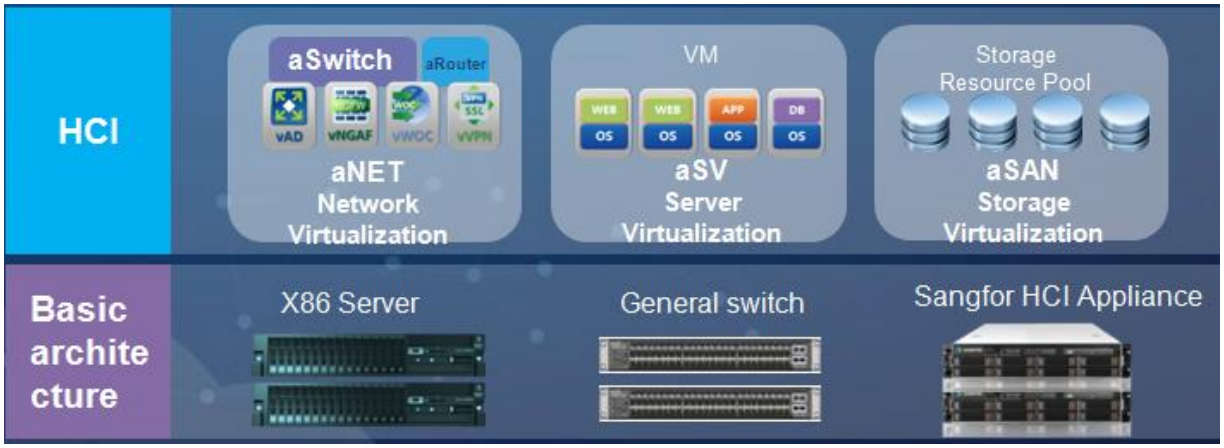
- An overview of cloud computing and cloud platform, and a guide on reading this document
- The major functionality of HCI technology
- Introduction to Sangfor
- The core values that Sangfor brings to customers
- Practice and trial of Sangfor

2. Sangfor HCI Solution

2.1 Hyper-Converged Infrastructure (HCI) Overview

Hyper-Converged Infrastructure (HCI) is an infrastructure that combines computing, networking and storage capabilities onto industry-standard x86 servers by using virtualization technologies. It is often predefined according to system requirements. All the resources are aggregated into a resource pool on node basis that is easy to scale out.

2.2 HCI Architecture of SANGFOR



Sangfor Hyper-Converged Infrastructure

As show in the above figure, the principle components of Sangfor HCI are: x86 hardware, switches and virtualization layer that virtualizes computing, storage, networking and security with the software aSV, aSAN and aNet respectively, previsioning full virtualization and all resources for the data center. In later sections, we shall discuss the technology of aSV, aSAN and aNet in details.

2.3 aSV (Server Virtualization)

2.3.1 aSV Overview

This is a virtualization software used for virtualizing computing capacity of x86 servers, enabling customers to create a wide variety of standardized virtual machines, which are like computers provided by one manufacturer, with same hardware configuration or driver as wanted.

Definition of Virtual Machine:

A virtual machine is a type of computer application used to create a virtual environment, which is referred to as "virtualization". Some types of virtualization let a user run multiple operating systems on one computer at the same time. A virtual machine can also function for a single program, allowing that one application to function in an isolated way. Users can setup multiple computers to function as one through virtualization, allowing the system to draw on greater resources than might otherwise be available.

Virtual Machine vs. Physical Server:

A virtual machine is not made of physical electronic components, but is composed by a set of virtual components (files), which have nothing to do with hardware configuration. In addition to the differences in composition, a virtual machine is superior to a physical server in the following:

Abstraction Decoupling

- ☑ Operational on any x86 server
- ☑ Upper-level application system is operational without making any change

Isolation

- ☑ Running is independent from other virtual machines
- ☑ No interaction among data processing, networking and data storage

Great Mobility after Encapsulation

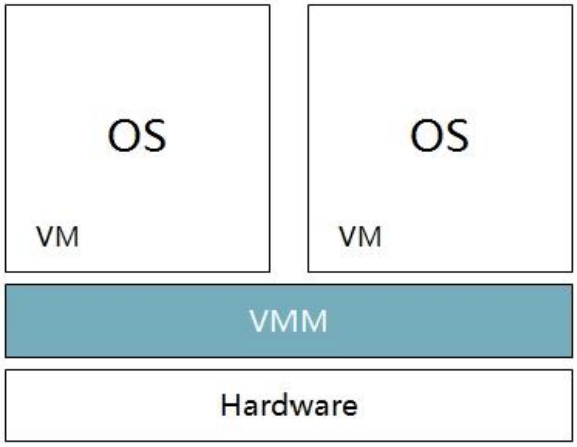
- ☑ Virtual machines are encapsulated into one file that is easy to duplicate, deploy, back up and recover
- ☑ Configuration, operation system and application programs of a virtual machine can be hot migrated as a whole among the hosts.

Sangfor consists of an aSV virtualization platform that virtualizes physical servers and creates more than one virtual machines. Users may install software, mount disks, change configuration and re-connect it like any ordinary x86 server.

Virtualization plays a crucial role in Hyper-Converged Infrastructure (HCI). For end users, virtual machine is superior to physical server in distribution, configuration change and networking. For IT operators, virtual machines can reuse hardware resources, and reduce overall IT investment and operational costs in conjunction with cloud automation capability.

2.3.2 aSV Technical Principle

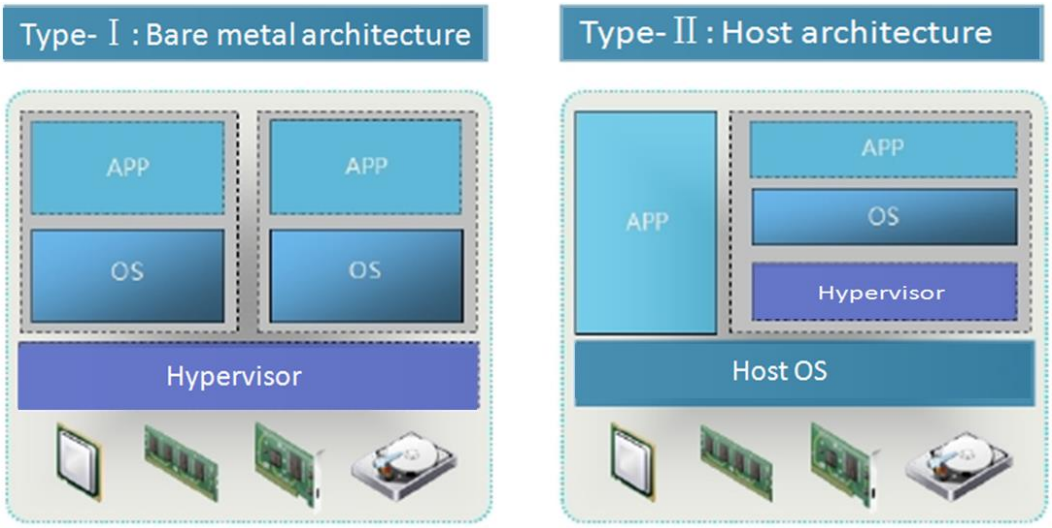
2.3.2.1 **Hypervisor Infrastructure**



A hypervisor, or VMM (Virtual Machine Monitor), is a piece of software running amidst the physical server and host OS, which enables more than one operating systems and applications share a set of physical hardware, coordinates accesses to physical components and virtual machines, creates and runs virtual machines.

Hypervisor is the core of virtualization technology. Its fundamental capability is to perform hot migration without interrupting any task. Upon startup of server and hypervisor, every virtual machine will be allocated with appropriate resources, such as memory, CPU, networking capability and disks, while guest operating system are loaded accordingly.

There are two types of hypervisors: Type-I (Bare-Metal) and Type-II (Host OS-Based)



Virtualization Infrastructure

Type-I (Bare-Metal)

This type of hypervisor runs directly on top of the host hardware. To access to the host hardware, Guest OS must have the interrupts from the hypervisor which uses the driver programs and acts like the immediate manipulator. This layer runs one or more operating system instances, which is virtualized to some extent.

Type-I hypervisor is a lightweight OS that manages and calls hardware resources, without relying on host operating system. Performance of this type of hypervisor is between server virtualization and OS-level virtualization.

Common examples of type-I hypervisor include VMware ESX Server, Citrix XenServer, Microsoft Hyper-V and Linux KVM, etc.

Type-II (Host OS-Based)

This type of hypervisor relies on a host OS. Guest OS requires host OS to access host hardware, which brings additional performance costs but makes the best use of the drivers and services provided by the host OS to manage memory and resources, and schedule process.

The VM application program of server virtualization requires the following when accessing to host hardware: VM kernel > hypervisor > host kernel. For that reason, this type of virtualization provisions the lowest performance.

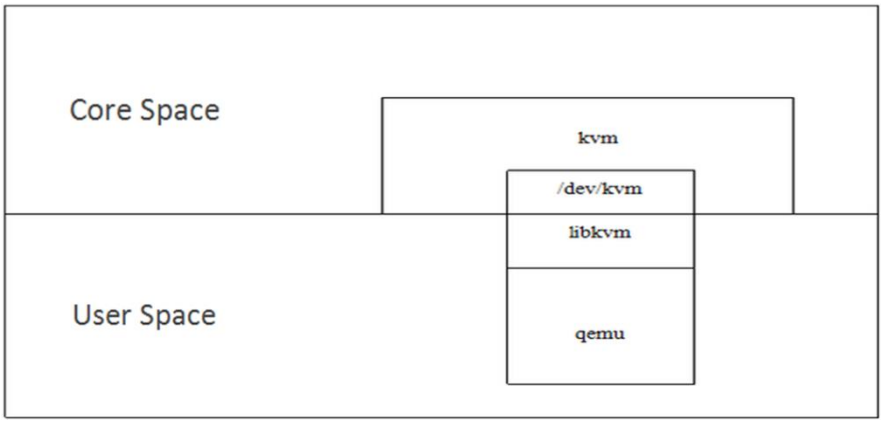
Common examples of Type-II hypervisor includes VMware Server(GSX), Workstation and Microsoft Virtual PC, Virtual Server, etc.

Having taken into account both performance and efficiency, Sangfor has chosen Type-I (Bare-Metal) hypervisor and Linux KVM.

KVM (Kernel-based Virtual Machine) is a full virtualization solution for Linux on x86 system that is integrated in each released version since Linux2.6.20. It makes use of the Linux scheduler to perform management and requires less crucial source code. KVM contains virtualization extension (Intel VT- X) and is revision of QEMU. It is a virtualization architecture for the Linux kernel and can be loaded with the modprobe command, and then create virtual machine with the corresponding tools. The KVM will accomplish nothing without kernel's help, for it requires a running userspace tool.

Sangfor chooses the mature and open-source QEMU hypervisor that can virtualize various types of processors. For example, with one physical processor on an x86 physical server, user can gain a virtual processor on top of that power, with which another CPU can be encoded to encode a program running on top of it.

KVM makes use of part of QEMU and improves it before making it userspace tool that can control KVM. That is the relation between KVM and QEMU is shown in the following figure:



An ordinary process in Linux can be executed in Kernel mode or User mode.

KVM supports a third mode, Guest mode, in addition to Kernel mode and User mode. In KVM model, each virtual machine is a standard process managed by Linux scheduler.

In conclusion, KVM consists of two parts, one is the hardware driver that manages configuration of virtualized devices, which takes ***/dev/kvm*** as the management interface; the other is the userspace component emulating PC hardware, a QEMU process that is revised a little bit.

More benefits of aSV with KVM:

- ☑ Being integrated into Linux released version to improve compatibility

- ☑ Source code-grade resource scheduling improves performance
- ☑ Virtual machine is like a process whose memory is easy to manage
- ☑ Adopting NUMA technology to improve scalability
- ☑ Source codes stay open to have more supports from community

2.3.2.2 **Implementation of aSV Hypervisor**

VMM (Virtual Machine Monitor) virtualizes three types of physical resources, namely, CPU, memory and I/O device, among which CPU virtualization is the key.

Classical Virtualization: In modern computing system, there is generally two privilege levels (User state and Kernel state, or four privilege levels Ring0 ~ Ring3 on x86 system) to isolate system software and application software. The privileged instructions are those highest privileged instructions in CPU in Kernel mode that can read or write commands for key resources. On x86 system, there are some sensitive commands are not privileged.

If some privileged instruction executions are not performed in Kernel state, it may incur an exception that is then handled (trapped) by system software as unauthorized access.

The classical way of virtualization uses “de-privilege” and “trap-and-emulate” to make guest OS run in non-privileged level, while VMM runs in the highest privileged level (having full control of the system resources). Once guest OS is de-privileged, majority of the guest instructions are executed by the hardware without being trapped and emulated by VMM unless a privileged instruction is executed.

Trap-and-emulate implementation style is to make VMM emulate the trapping instructions that may affect VMM operation, but most of the insensitive instructions are executed as usual.

For x86 architecture, there are more than one instructions are sensitive

that need to be handled by VMM, but they are not privileged. For those instructions, de-privileging does not make them trapped and emulated by VMM. As a result, they interrupt instruction virtualization. That is virtualization vulnerability.

x86 Architecture Virtualization Implementation Types

☑ x86 full virtualization

An abstracted VM owns all the attributes of a physical machine, but guest OS is not required to make any modification. The procedure of monitor is accomplished during the running process, and instructions are emulated after being trapped. Yet there are differences in implementation among varied virtualization approaches. VMware is a typical example that uses Binary Translation (BT), which translates guest OS instructions to subset of the x86 instructions set, while sensitive calls are set to automatically trap. The translation and instruction execution are performed simultaneously, while insensitive instructions in User mode are executed without being translated.

☑ x86 para-virtualization

Para-virtualization requires assistance of OS virtualization and modification of OS. It modifies the OS kernel to replace sensitive instructions with hypercall, similar to the OS calls to move privilege to VMM. Para-virtualization is widely known due to its application in VMM.

This technology improves VM performance to the maximum, even close to that of a physical machine. Its drawback is Guest OS modification (not supported by Windows platform) and increasing maintenance costs. What's more, modified Guest OS will become dependent on specific hypervisor. For those reasons, many virtualization solution suppliers have given up Linux para-virtualization on VMM based virtualization solution, but turn

to and focus on hardware assisted full virtualization to support unmodified OS.

☑ x86 hardware assisted virtualization

The virtualization concept is to introduce new instructions with a new CPU execution mode that allows the VMM to run in a new root mode different from Guest OS. The Guest OS runs in Guest mode, where privileged and sensitive instructions are set to automatically be trapped into the VMM, the difficulty of trapping less privileged instructions is thus removed. For every mode change, contexts are saved and restored by hardware, which dramatically improves the efficiency of context switches during the trap-and-emulate process.

Take hardware assisted virtualization technique based on Intel VT-x for example, it increases two modes in processor in virtualized state: Root mode and Non-root mode. VMM runs in Root mode, while Guest OS runs in Non-root mode. Both modes have privileged ring. VMM and Guest OS run in ring 0 in the two modes respectively. In that way, VMM is able to run in ring 0, so is Guest OS without being modified. Switches between Root mode and Non-root mode are accomplished by adding CPU instructions, such as VMXON, VMXOFF.

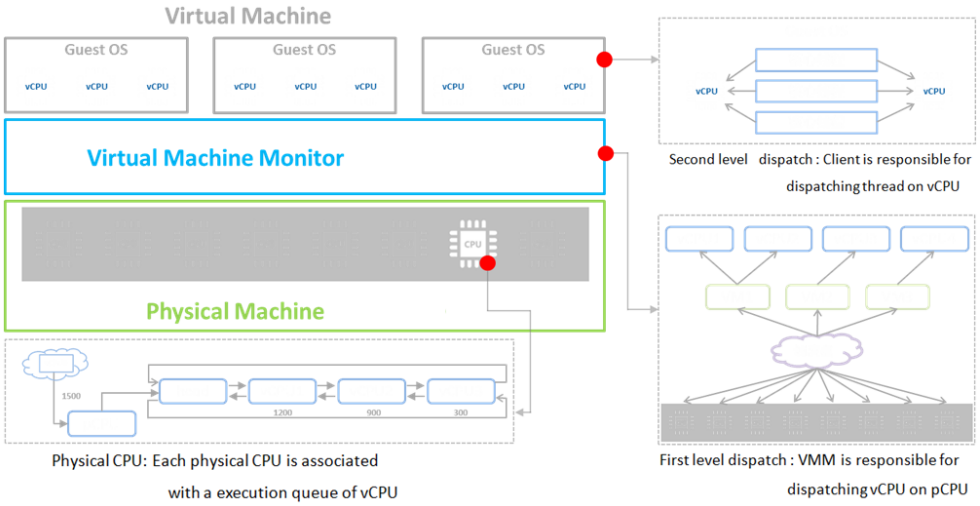
Growing number of virtualization solution suppliers embrace hardware assisted virtualization, as hardware assisted virtualization technique eliminates the OS' ring switch problem and makes virtualization simpler but requires no modification on OS kernel. Hardware assisted virtualization is gradually reducing the differences between various software virtualization techniques and developing as the trends.

vCPU Scheme

Guest is unaware of the physical CPU but aware of the vCPU through which processing unit is exhibited.

In VMM, every vCPU has a VMCS (Virtual-Machine Control Structure) where vCPU is switched away from or to the physical CPU and contexts are saved in or imported to the physical CPU. Through that way, vCPUs are separated from one another.

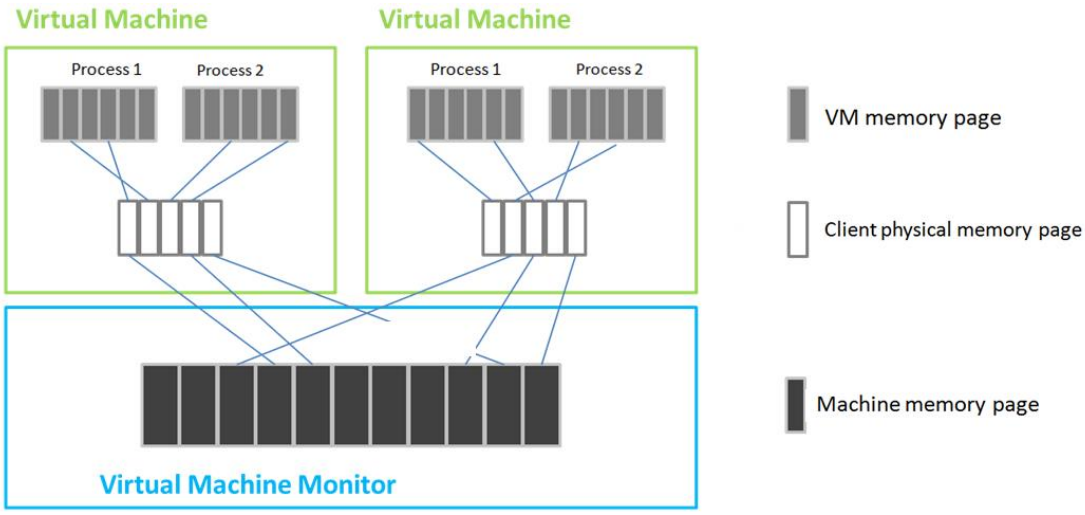
From VM system's structure and function separation, we know that Guest OS and VMM make the two-level scheduling scheme of a virtual machine system. The following figure shows a VM scheduling scheme in multi-core environment. Guest OS is in charge of level 2 scheduling, i.e., thread or process scheduling on vCPU (core threads being mapped to the corresponding virtual CPU). VMM is in charge of level 1 scheduling, i.e., vCPU scheduling on physical CPU. The policy and scheme in the two levels of scheduling are independent.



vCPU Scheduling Mechanism

vCPU scheduler allocates and schedules the physical processors among virtual machines, catering the varied resource needs from all virtual machines based on specific policy or scheme to schedule physical resources from the perspective of physical processing unit. It could schedule execution of one or more physical processing units (reused in different time or locations), or build mapping with one physical processing unit (to restrain the accessible physical processing units).

Memory Virtualization



Three-Layer Mode of Memory Virtualization

VMM (VM Monitor) manages all the system resources, including memory resources, page memory, mappings from virtual addresses to the guest memory physical addresses. Guest OS itself provides page memory management mechanism, therefore VMM owns one more mapping relation than ordinary system:

a. Virtual Address (VA) is the liner space provided for other application programs by the by Guest OS
b. Physical Address (PA) is abstracted guest physical address
c. Machine Address (MA) is real address

Mapping relation is as follows: Guest OS: PA = f (VA), VMM: MA = g (PA)

VMM manages a set of page tables that contain mappings from physical addresses to machine addresses. Guest OS manages a set of page tables that contain mappings from virtual addresses to physical addresses. In operation, user program accesses VA1 that is translated to PA1 by page table of the Guest OS, and then VMM gets involved and translates PA1 to MA1 according to the Guest OS page table.

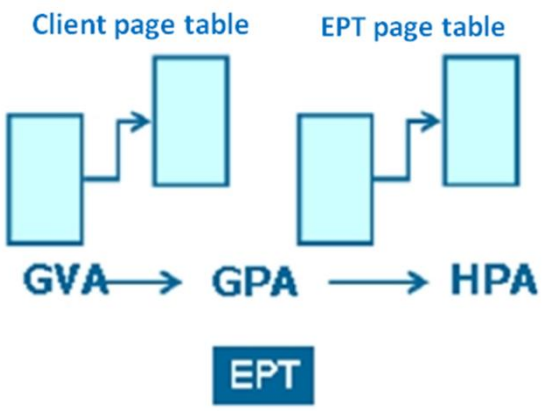
Page Table Virtualization Technique

Conventional MMU translates virtual addresses to physical addresses only once. However, in virtualized environment, the so called physical address is actually not the real machine address. To obtain the real machine address, VMM must get involved and another mapping is required. Address mapping is impractical as the efficiency is extremely low if every memory access by virtual machine requires VMM's involvement. The most common way to translate virtual address to machine address is to use VMM to generate mapping fg according the mapping of f and g, and then write the mapping to MMU. Page table virtualization technique mainly uses MMU paravirtualization and shadow page table to accomplish virtualization. The latter has been replaced by hardware assisted hardware virtualization.

☑ MMU paravirtualization

For this virtualization technique, the Guest OS will allocate a page to the new page table when it constructs a new page table, and then register it at VMM. The VMM de-privileges the Guest OS to write the page table, since when VMM will intercept traps if guest OS writes the page table, and verify and translate the address. VMM verifies every item in the page table, making sure they are mapping the machine pages related to the corresponding virtual machine, not containing the writable mappings against the page table page. According to its own mappings, the VMM translates the physical addresses to machine addresses, and then load the modified page table to MMU. By that means, the MMU can translate the virtual addresses to machine addresses.

☑ Memory hardware assisted virtualization



Working principle

Hardware assisted virtualization of memory is a technique designed to replace the shadow page table in software virtualization. The basic working principle is that two address translations are executed by CPU automatically, from GVA (Guest Virtualization Address) to GPA (Guest Physical Address) and to HPA (Host Physical Address), rather than by software which requires large memory and is lower in performance. Take VT-x Extended Page Tables (EPT) for example. VMM first translates the physical address of the virtual machine to EPT page table and sets it to CPU, and then the virtual machine modifies the guest page table without the VMM's interrupt. Lastly, when address is being translated, the CPU automatically finds the two page tables to complete translation from virtual address to machine address.

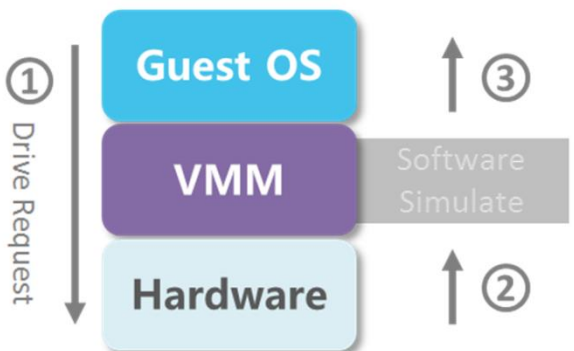
In conclusion, memory hardware assisted virtualization does not need VMM's interrupt during guest operation and saves guest OS a lot workloads, performance is as high as that of a physical machine.

I/O Device Virtualization

To reuse the limited I/O devices, VMM must virtualize the I/O devices. This is accomplished by intercepting, trapping and emulating all the I/O operations issued by the guest operating system.

Currently, there are three types of approaches for I/O devices virtualization, namely Full Emulation of Network Port, Front-end/Back-end Driver Emulation and Direct Allocation.

☑ Full emulation of network port

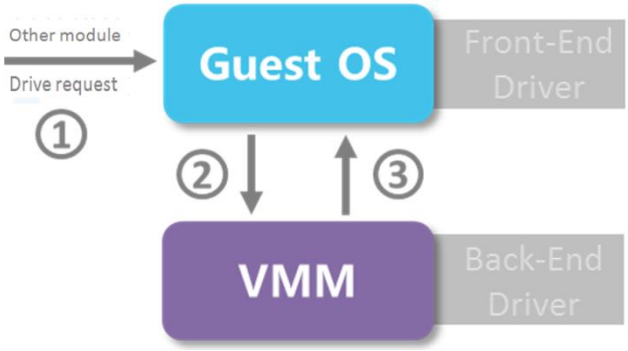


Software mimics the physical port and emulates one that is exactly the same with the physical device. Guest OS does not need to be modified at all to run the virtual machine.

Benefit: It does not need any additional hardware cost as the existing drivers can be reused.

Drawback: Every operation involves in more than one registers, and every access to a register must be intercepted by VMM and emulated accordingly. For that reason, switching occurs frequently. What is more, software emulation is generally lower in performance.

☑ Front-end/back-end driver emulation



VMM provisions simplified drivers (back-end driver and front-end driver). Guest OS driver is Front-End (FE), which is used to send request from other modules to back-end driver through the special

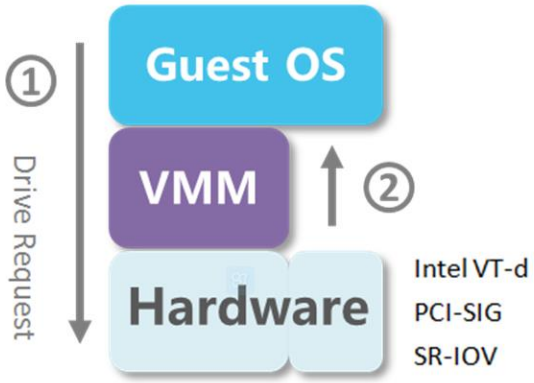
communication scheme with Guest OS. Once back-end driver processes the request, it responds the front-end driver.

Benefit: Such business-based communication scheme can reduce switching costs to the minimum, and no additional hardware cost is required.

Drawback: Back-end driver may become bottleneck as Guest OS needs to accomplish front-end driver.

☑ Direct allocation

Physical resources are allocated to a specific Guest OS directly. Guest OS has direct access to the I/O device, without interrupts from VMM. Some technologies are meant to establish efficient I/O virtualization channel, such as IOMMU (Intel VT-d, PCI-SIG SR-IOV, and so on).



Benefit: The existing driver can be reused, and therefore virtualization cost is reduced due to direct access.

Drawback: It requires additional hardware investment.

2.3.3 aSV Technical Features

2.3.3.1 **Memory NUMA Technology**

Non-uniform memory access (NUMA) is a new server CPU and memory design architecture. Under the traditional server architecture, the

memory is put into a single storage pool, which works well for single-processor or single-core system. However, this traditional way of universal access will cause resource contention and performance issues when multi-cores access the memory space at the same time. After all, CPU should be able to access all of the server memory, but do not always remain occupied. In fact, CPU only needs to access the memory space required by workload when actual running.

Therefore, NUMA has changed the way memory presents CPU. This is achieved by the partition of each CPU memory of the server. Each partition (or memory block) is called NUMA node, and the processor associated with this partition can access NUMA memory faster, and does not need to compete with other NUMA nodes for resources on the server (other memory partitions are allocated to other processors).

The concept of NUMA is associated with cache. Processor speed is much faster than memory, so the data is always moved to a faster local cache, where the processor access speed is much faster than the common memory. In essence, NUMA configures a unique overall system cache for each processor, reducing contention and delay when multi-processors try to access a unified memory space.

NUMA is fully compatible with server virtualization, and can also support any processor to access any piece of memory space on the server. Of course, a processor can access the memory data located on different areas, but it requires more than the transmission of the local NUMA node, and requires the confirmation of target NUMA node. This increases the overall costs and affects the performance of the CPU and memory subsystem.

There are no compatibility issues when NUMA load virtual machine, but in theory the perfect way of virtual machine should be within a NUMA node. This prevents the processor from needing to interact with other NUMA nodes, which results in the decrease of workload performance.

Sangfor aSV support NUMA technology, making the hypervisor and upper OS memory interconnect, so that OS would not migrate workloads between the CPU and NUMA node.

2.3.3.2 **SR-IOV**

Typically the technologies for server virtualization satisfy the I/O requirements of virtual machines by software simulation sharing and a physical port of virtualization network adapter. The multiple layers of simulation make I/O decisions software for virtual machines, leading to environment bottlenecks and affecting I/O performance. SR-IOV provided by aSV virtualization platform is a method which can share the physical features of I/O device and I/O ports without software simulation, mainly utilizing iNIC to achieve bridge unmount the virtual card and allowing to allocate the SR-IOV virtual function of physical network adapter directly to virtual machines. Therefore, SR-IOV can improve network throughput, reduce network latency, and reduce the required host CPU overhead to handle network traffic.

Technical Principle: SR-IOV (Single Root I/O Virtualization) is a standard launched by PCI-SIG, is a technical realization of the virtual channel (virtualize multiple physical channels for the upper-layer software systems on the physical NIC, each with independent I/O functions), and is used to virtualize a PCIe device into multiple virtual PCIe devices, with each device provides the same services to the upper layer software as a physical PCIe device do. Through SR-IOV, a PCIe device can export not only multiple PCI physical functions, but also a set of virtual functions of the resources shared on the I/O device. Each virtual function can be assigned directly to a virtual machine, allowing network transmission to bypass the software emulation layer and be directly assigned to a virtual machine, therefore achieves the object of assigning the PCI function to multiple virtual interfaces to share a PCI device in the virtual environment and reduces the I/O overhead of

software simulation layer, so as to achieve a near-native performance. As shown in the figure, no transparent transmission is needed in this model, because virtualization occurs on the terminal device, allowing the management program to simply map virtual functions to the VM to achieve the device performance and isolation safety of native machine.

The channels virtualized by SR-IOV include two types:

- ☑ PF (Physical Function) is a complete PCIe device that contains a comprehensive management and configuration capabilities. Hypervisor uses PF to manage and configure all I / O resources of the NIC.
- ☑ VF (Virtual Funciton) is a simplified PCIe device which contains only I/O functions. VF is derived from PF and appears to be a slice of physical NIC hardware resources. For Hypervisor, this VF is exactly the same as an ordinary PCIe card.

By using SR-IOV, we can meet high network IO application requirements, without having to install special drivers and affecting hot migration, memory multiplexing, virtual machine network control and other virtualization features.

2.3.3.3 **Faik-RAID**

Under normal circumstances, when the host system has more than one hard disk, it is often considered as first option to improve disk performance or provide disk redundancy by forming Raid. Today's mainstream raid implementations can be broadly divided into three types:

- ☑ Hardware raid (hardware raid): achieved by purchasing expensive raid card.
- ☑ Software raid (software raid): create an array through the

operating system software, and CPU is responsible for the processing overhead of raid.

- ☑ Motherboard raid (fake raid): create arrays through building raid controller within the motherboard, which is identified by the operating system drivers.

With respect to the expensive hardware, motherboard raid (fake raid) would be a good choice for us. Fake raid only provides cheap controller, and raid CPU processing overhead is still handled by CPU, so the performance and CPU usage are basically about the same with software raid.

aSV 3.7 incorporates support for Fake-RAID installation and use of Fake-RAID storage. Now Intel mode raid0, raid1, raid5, raid10,and LSI model raid0 can all be used on aSV 3.7.

2.3.3.4 **Life Cycle Management of Virtual Machines**

aSV provides a comprehensive management of the entire process of virtual machines from creation to deletion. Just like the human life cycle, the basic life cycle of virtual machine is three states: creation, use and deletion. Of course it also includes the following states:

- ☑ Create VM
- ☑ VM power on and power off, reboot and suspend
- ☑ Operating system installation on VM
- ☑ Create template
- ☑ Update VM hardware configuration
- ☑ Migrate VM or VM's storage resource
- ☑ Analyze the resource utilization of VM

- ☑ Backup VM
- ☑ Restore VM
- ☑ Delete VM

In the life cycle of a virtual machine, it may experience these states at a certain time point. aSV provides complete virtual machine lifecycle management tools, through which we can plan on a virtual machine lifecycle and maximize the role of the virtual machine.

2.3.3.5 **VM Hot Migration**

In virtualized environment, physical servers and storage carry much more business and data; therefore can suffer more from equipment failure. aSV virtualization platform provides virtual machine hot migration technology to reduce the risk of downtime and business interruption time.

aSV virtual machine hot migration technology refers to migrate a virtual machine from one physical server to another physical server, namely virtual machine save/ restore. First, the overall state of the virtual machine is preserved intact, and can be quickly restored to the target hardware platform, whereas the virtual machine is still operating smoothly after recovery and the user will not notice any difference. Virtual machine hot migration technology is mainly used for dual fault-tolerant, load balancing, energy saving and other scenarios. aSV virtualization platform hot migration provides memory compression technology to double hot migration efficiency and support up to four concurrent virtual machines to migrate.

Values:

- ☑ In the maintenance process, migrate application to another server manually through hot migration, and then migrate back

after maintenance. In the meantime, the application would not stop, thus reduce planned downtime.

☑ Combine with dynamic resource scheduling strategies. For instance, when the virtual machines' load is reduced at night, it will automatically migrate virtual machines to concentrate to part of the servers through preconfiguration to reduce the number of running servers, thus reduce energy expenditures on equipment operation.

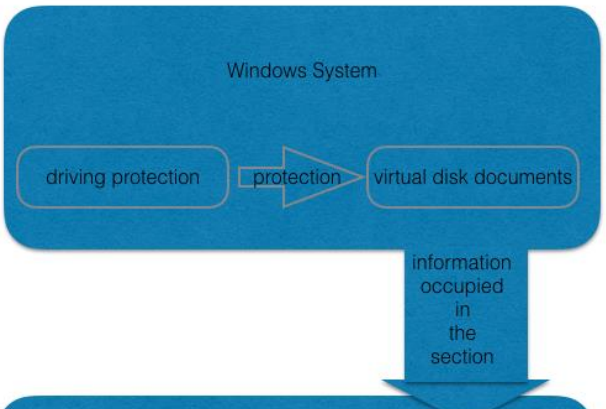
2.3.4 aSV Special Technologies

2.3.4.1 Fast Physical Machine Virtualization

In actual IT infrastructure virtualization, migrating Windows based application system to a virtualized environment is an inevitable requirement. Instead of using conventional P2V or V2V tools to convert physical machines to virtual machines, Sangfor uses fast conversion tool to convert Windows servers, integrated with innovative conversion technology.

The following introduces the principle of P2V conversion: On a Windows computer to be converted, a virtual disk file will be created in a physical sector and protected by Windows driver to ensure that the physical sector will not be moved; and information of that physical sector will be obtained and saved to the system configuration file; then boot program and kernel of Sangfor will be installed, and a boot item will be added to the system so that the computer can boot from the operating system of Sangfor . When data is read from or written to the virtual disk on Sangfor platform, the virtual disk driver will read data from or write data to the physical sector where that virtual disk file is stored. When the computer enters operating system of Sangfor again, that piece of data can still be read and written.

With Sangfor Converter, physical machines could be quickly converted to virtual machines and hypervisor created on those physical servers.



2.3.4.2 **High Availability**

High Availability is HA for short. If HA is enabled on a virtual machine, that virtual machine will be migrated to another node in case that network cable is dropped or storage is inaccessible, etc., so as to ensure service continuity.

Virtual machine status will be checked every 5 seconds. Once any fault occurs, that virtual machine will be migrated to another node.

HA will be triggered if any of the following occurs:

- ☑ It has been detected three times that the physical NIC on the node that a virtual machine is running on is unplugged (exclusive of the situation with NIC disabled)
- ☑ It has been detected twice that the node on which the virtual machine is running cannot access the datastore of that VM.

With use of HA, downtime due to node or link failures could be reduced greatly.

2.3.4.3 **Resource Scheduling**

In virtualization environment, if a business system is installed on a virtual machine that runs on a node without enough available resources, resource needs of that VM may not be met, which may affect performance of that business system.

With help of resource scheduling, resource distribution could be automatically balanced among nodes. If a virtual machine runs low on

resources, it could be migrated to another node with more available resources so as to ensure all applications on it operate properly.

With resource scheduling enabled, it is possible to run lots of virtual machines that require a high CPU and memory usage, such as a virtualized database server, since resources could be automatically scheduled among nodes, which can greatly reduce operational costs.

When a virtual machine lacks resources, resource scheduling performs virtual machine migration based on migration-triggering rule, and CPU and memory usage of clustered hosts monitored regularly using host heartbeat mechanism. Then, that VM may be migrated to another node with more available resources, or other VMs may be migrated to another node.

2.3.4.4 **Multiple USB Devices Mapping**

If an application on a physical server, such as Kingdee, needs to be encrypted via a USB key, the USB key should be plugged into that server after the server is virtualized, and mapped to that virtualized server. More requirements may include mapping during hot migration and migration across hosts

Currently, there are three solutions available:

- ☑ Host mapping: This solution does not support network mapping. Therefore, hot migration cannot be supported.
- ☑ Anywhere USB: This solution utilizes an IP-based intermediate device, installs driver on the virtual machine and configures peer device.
- ☑ Hardware virtualization and proxy: This solution supports network mapping and hot migration without any changes to the guest OS. When a physical machine is converted to a virtual machine, the VM can directly read data from and write data to a USB device mapped to that VM. This solution eliminates the drawbacks of other two solutions and the problem of using USB device in virtualization environment.

How hot migration is implemented: Since the communication between a virtual machine and USB device is over network, a message will be sent to tell USB service program to change IP address of the destination node when the virtual machine is migrated to another node, and then a connection to the new destination node will be initiated. Once the new connection is established successfully, the communication with USB device will be recovered, which is transparent to the virtual machine.

Sangfor platform utilizes the third solution which supports mapping multiple USB devices and provides the following benefits:

- ☑ Give prompt once a USB device is detected.
- ☑ There is no need to install any plugins on virtual machines.
- ☑ Support mapping USB device across nodes and hot migration so as to accommodate cluster environment.
- ☑ Automatically mount previously-mapped USB device to virtual machine after migration.
- ☑ Provide a virtual device functioning like a USB Hub, which works with Sangfor platform for establishing a USB device mapping environment.
- ☑ Re-map USB device to the virtual machine when it recovers from failure, such as reboot due to fault occurrence, or network connection error on the node to which that USB device is attached.

2.4 aSAN (Storage Area Network)

2.4.1 Storage Virtualization Overview

2.4.1.1 Challenges for Storage after Virtualization

By utilizing virtualization technology, utilization of resources on servers becomes higher, business system becomes easier to be deployed and

total cost of ownership (TCO) becomes lower as well, but challenges come along.

Compared with traditional solution using physical servers, a storage system serves more services and therefore requires a higher performance.

With use of shared storage, tens or hundreds of virtual machines may run on a volume, leading to random characteristics on volume IO, which is a challenge to traditional Cache technology.

To make more than one virtual machines run on a same volume, storage system must be able to coordinate access requests from those virtual machines, ensuring that virtual machine with a high IO throughput can access resources preferentially.

It requires a high IO performance to support virtual machines running on a same volume, which is also a challenge to the traditional solution using RAID technology.

2.4.1.2 **Development of Distributed Storage Technology**

Typical distributed storage generally falls into the following types: file-based storage, object-based storage and block-based storage. Distributed storage technology becomes mature and is widely used in IT industry, which is proved by Server SAN and other related products. For example, use of distributed storage tech is applied to search engine and public cloud, since it has the following benefits:

- ☑ **High performance**: Data is distributed among servers so as to achieve load balancing.
- ☑ **High reliability:** A single point of failure (SPOF) no longer exists in the cluster, since multiple copies of data could be configured as needed and stored on different servers, hard disks and nodes on different racks. Thus, service will not be interrupted even

though one node fails, because the copy of data on that failed node can be automatically reconstructed.

- ☑ **High scalability:** Storage node can be increased linearly without limit to storage capability.
- ☑ **Simple management:** Storage software is directly deployed on servers and no dedicated physical storage device is required. What's more, storage can be easily configured, managed and maintained via web-based access.

2.4.1.3 Overview of Sangfor aSAN

aSAN is a distributed storage solution provided by Sangfor to meet requirements of storage virtualization and a critical component of hyper-converged infrastructure. It is developed based on the distributed file system GlusterFS and designed for cloud computing environment. Additionally, it is integrated with other features, such as distributed cache, SSD read/write caching, data redundancy and auto data reconstruction after fault occurrence, etc., which meets storage requirements of crucial services and ensures that services operate steadily, reliably and efficiently.

2.4.2 aSAN Working Principle

All the hard disks in the cluster are managed by aSAN running on hypervisor, with help of host and disk management, storage area network, caching and data redundancy techniques. aSAN pools storage provided by hard disks in the cluster and provides an interface to Sangfor so that virtual machines can save, manage, write data to and

read data from the storage pool.

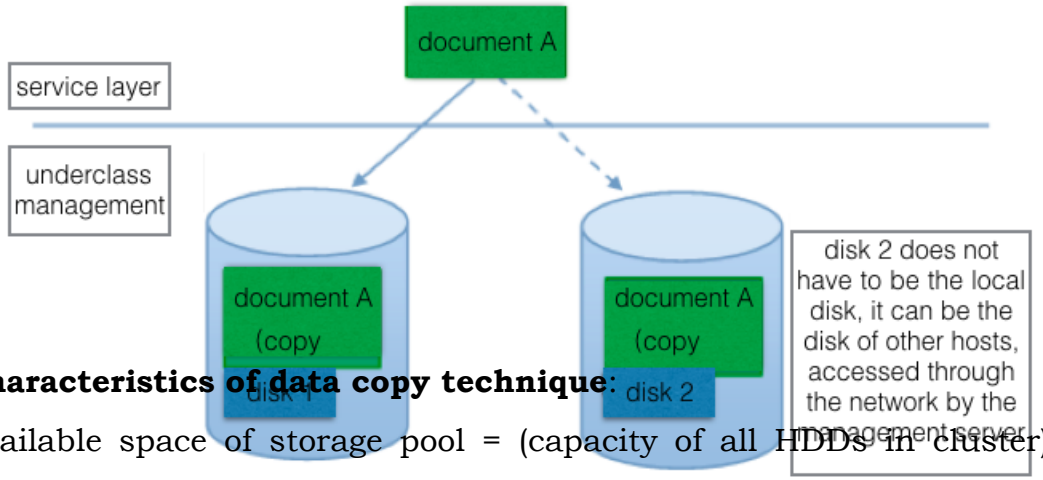
2.4.2.1 Node Management

Since aSAN needs to gather information of nodes in the cluster, the cluster must be created before establishing aSAN. Therefore, there must be at least two nodes in aSAN environment.

2.4.2.2 Data Copy

Gaining some knowledge about data copy technique is necessary before reading the next section on disk management and data copy configuration.

Data copy technique is used to store a same piece of data on multiple storage. Sangfor aSAN replicates data on file basis. For example, file A has two copies, i.e., that file is saved on both disk 1 and disk 2 simultaneously. The two copies of file A are always the same as long as failure does not occur.



Characteristics of data copy technique:

Available space of storage pool = (capacity of all HDDs in cluster)/ (number of copies) (for the situation that all disks are of the same capacity). Therefore, available storage space becomes less when number of copies increases.

Guest OS is unaware of copies of file. Disk management and copies of file distribution are handled by virtual storage. Note that aSAN replicates data on file basis.

Copies of file are always the same as long as no failure occurs. Therefore,

there is no primary copy of file and secondary copy of file.

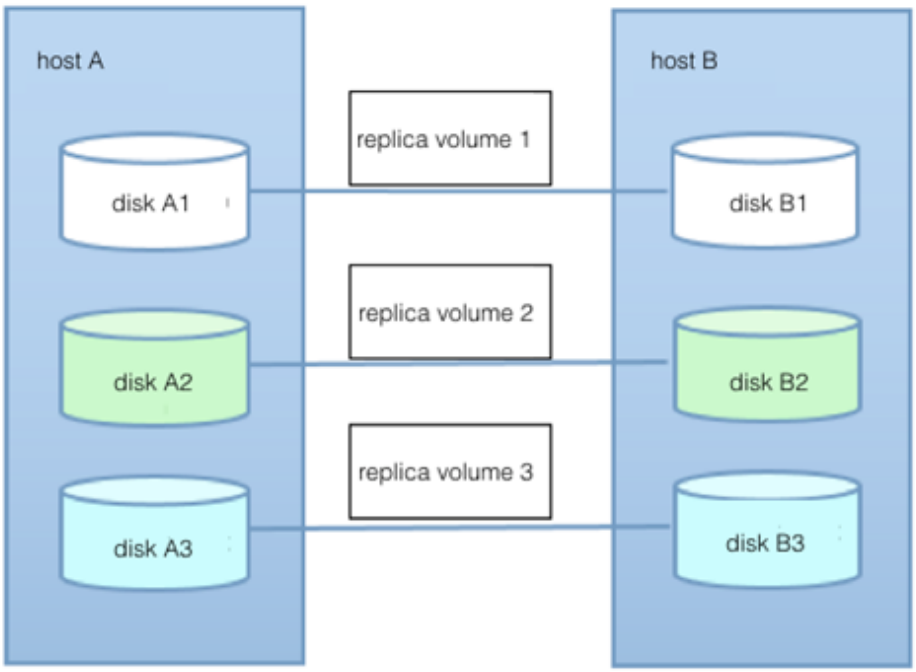
If any change is made to file A, which will be synchronized to the two copies of that file. For example, a piece of data is written to file A, which will also be written to the two copies of that file. However, a piece of data is read from one of the two copies of the file A.

2.4.2.3 **Disk Management**

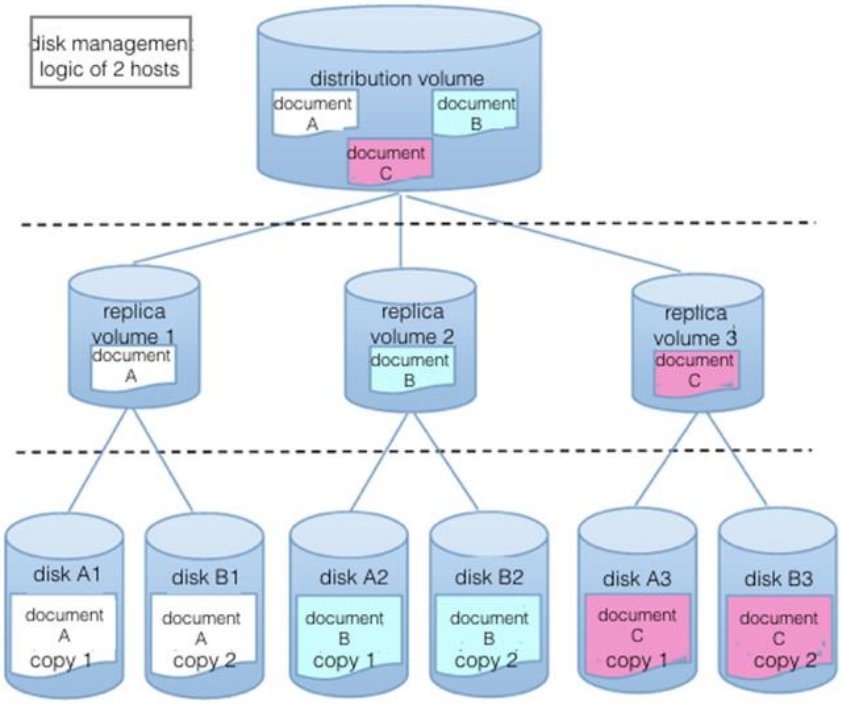
The policy of how to group disks is determined by aSAN disk management service based on number of nodes in the cluster and the number of copies of data specified during aSAN initialization.

aSAN disk management is provisioned in a multi-node cluster environment where each piece of data owns two or three copies stored on different nodes, to ensure data integrity in case of node failure. As data copy is across nodes, the algorithm for grouping data volume copies is the key.

Take the example of the scenario in the following diagram. In this scenario, there are two nodes with three disks on each node. And two copies of data are built across the two nodes.



When building two copies of data on the two nodes, disks on the nodes will be appropriately grouped to form three replicated volumes. The corresponding logical view is as shown below:

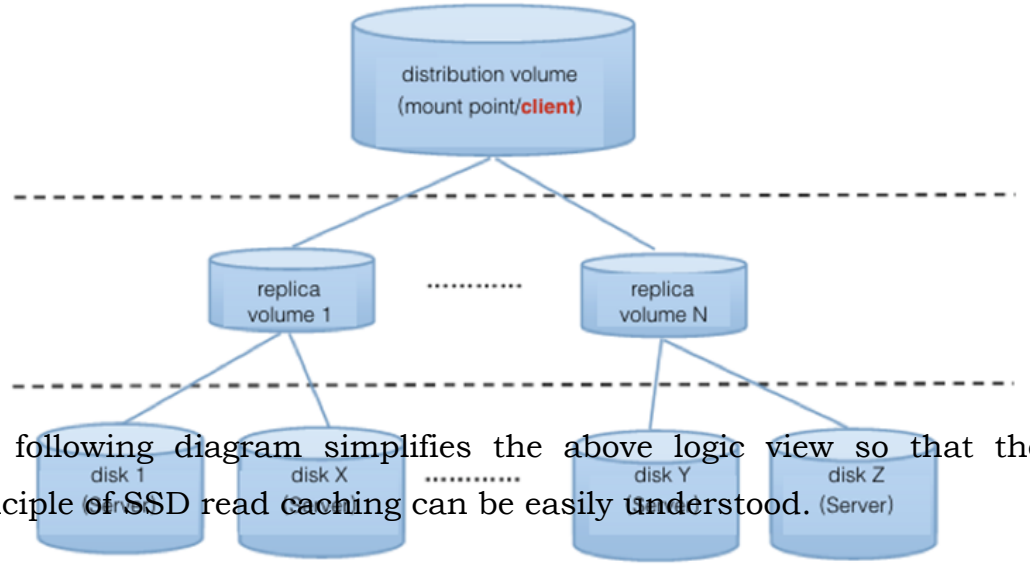


There is no essential difference between the diagram of the two nodes mentioned before and the diagram above. The only difference is that volume copies on physical disks are distributed to different nodes.

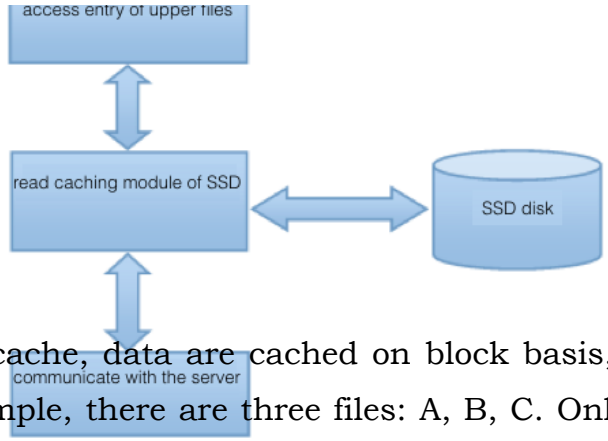
2.4.2.4 **SSD Read Caching**

In aSAN, a storage area network, SSD will be used for caching by default.

There are concepts of aSAN client and server. aSAN server handles disk IO requests, while aSAN client provides interface to virtual machines for accessing storage, such as a mount point. SSD read cache operates on aSAN client, while SSD write cache operates on aSAN server. The corresponding logical view is as shown below:



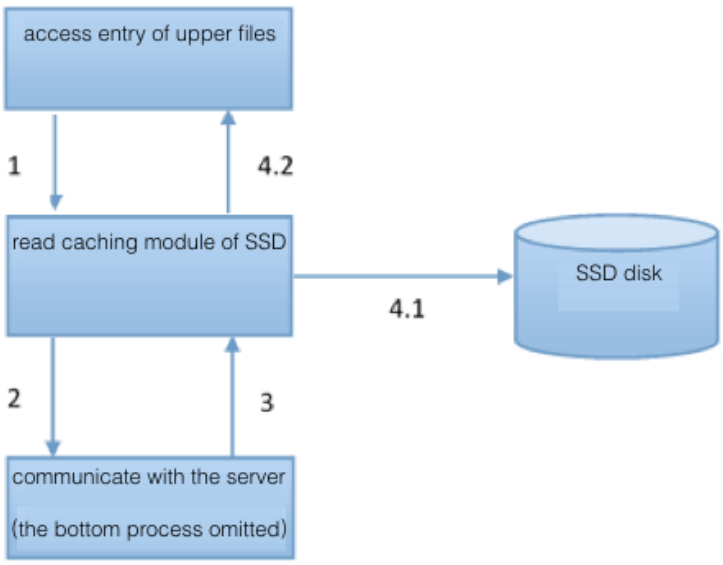
The following diagram simplifies the above logic view so that the principle of SSD read caching can be easily understood.



On SSD read cache, data are cached on block basis, rather than file basis. For example, there are three files: A, B, C. Only the data block which has ever been read will be cached in the corresponding files respectively.

SSD read cache module is between file access module and aSAN server. Therefore, all the SSD IO requests will go through and be handled by that read cache. The following introduce how file is read for the first time and for the second time respectively, and how the file is written.

Reading File for the First Time



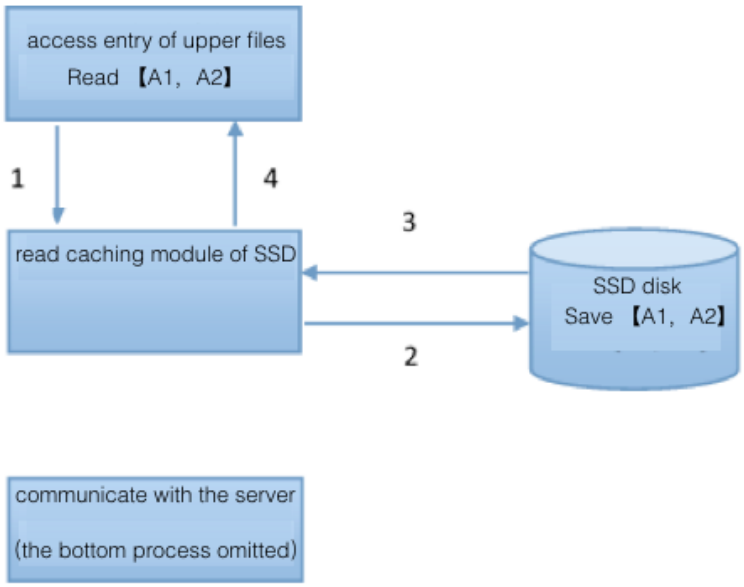
The following illustrate how the data that has never been cached is read for the first time:

- ☑ If the data block [A1, A2] of file A is requested, the read request is sent to SSD read cache module. Since the data is requested for the first time, it is not in the cache. Then, the request is passed to aSAN server.
- ☑ aSAN server responds to the request with the data block [A1, A2].
- ☑ The data block is returned to read cache module.
- ☑ This step consists of two other steps: 4.1 and 4.2. Returned data block goes through read cache module, it will be copied and saved to SSD, and a corresponding index will be created for this data block. This is Step 4.1. That data block will be returned to file access module simultaneously. This is Step 4.2. Step 4.1

and 4.2 are executed at the same time. Therefore, cache operation will not affect the read operation.

☑ The data block [A1, A2] is saved to SSD. If that data block is read again, it will be directly read from the SSD.

Reading File for the Second Time



The following illustrate how the data block that has ever been cached is read (assume that the data block [A1, A2] is cached to SSD):

☑ If the data block [A1, A2] of file A is requested, the read request is sent to the read cache module.

☑ Since this data block [A1, A2] is in the cache, the read cache module will initiate a read request to SSD to read that data block [A1, A2] .

- ☑ The data block is read from SSD and returned to the read cache module.
- ☑ The read cache module returns the data block to file access module.

Therefore, aSAN client can directly return the cached data block [A1, A2] to file access module without communicating with aSAN server, which reduces latency and IO workload of HDD.

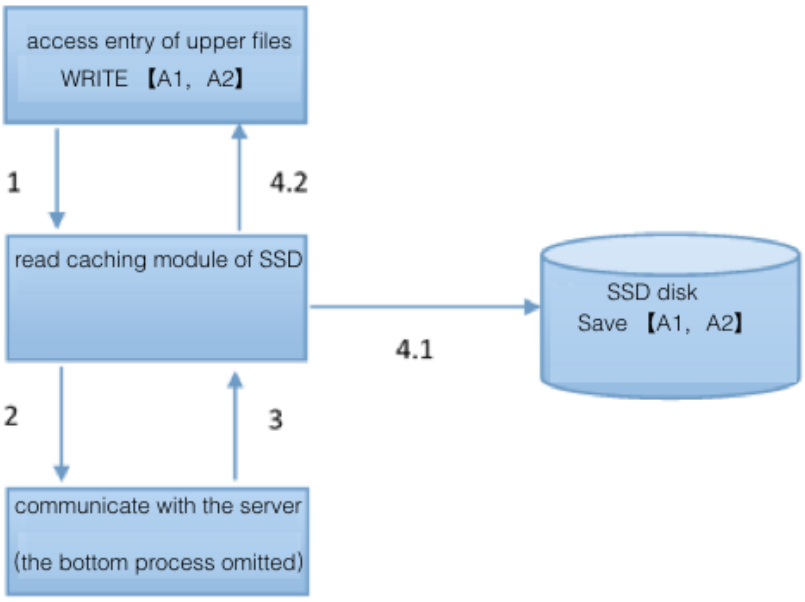
Writing File

To make the data in the read cache consistent with that on physical disks, a corresponding operation (such as update) will be executed by the read cache module while data is written to the physical disks.

The operation executed by read cache module is based on the principle that the recently written data is most likely to be read recently. For example, a certain file is uploaded to FTP server and will be most likely to be read in the near future.

The operation executed by read cache module for write operation falls into the following: writing data to SSD read cache, updating data in SSD reach cache.

1) Writing data to SSD read cache for the first write operation



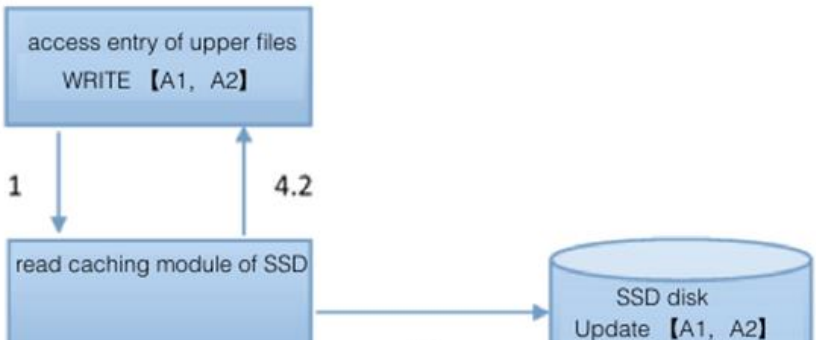
Following are illustration to the first write process, as shown in the figure above (assume that the data block [A1, A2] is written for the first time):

- ☑ The write request is transmitted through SSD read cache module, then passed to the aSAN server directly since it is a write request.
- ☑ The request is passed to the server and then data is written to HDD. After the write operation is complete, a result will be returned.
- ☑ The result is transmitted through the read cache module. If the result indicates the write operation is successful, it goes to Step 4; if it indicates the write operation is failed, the result will be directly returned to file access module without going to Step 4.
- ☑ This step consists of two other steps: 4.1 and 4.2. The data block [A1, A2] will be copied and saved to SSD by the read cache module, and a corresponding index will be created for this data block as well. This is Step 4.1. The result of the write operation will be returned to the file access module simultaneously. This is Step 4.2. Step 4.1 and 4.2 are executed at the same time. Therefore, cache operation will not affect the write operation.

Thus, the data block [A1, A2] is saved to SSD. The process of subsequent access to the data block is the same as that of reading file for the second time, speeding up data access.

2) Updating SSD read cache for the second write operation

Read cache module will update the data block which has been cached when it is written again.

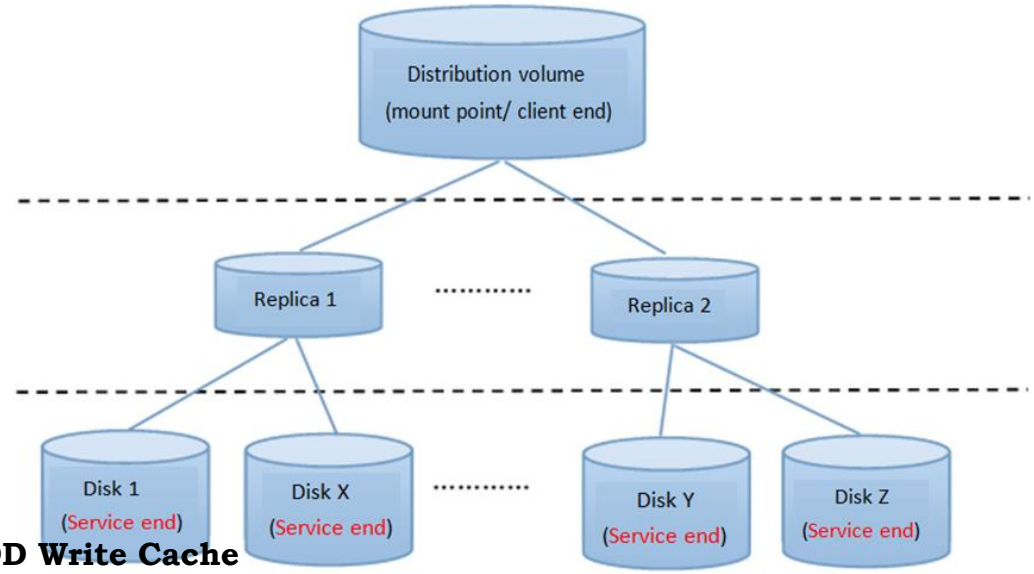


Assume that the data block [A1, A2] has been cached and a virtual machine initiates a write request (such as update) to that data block.

- ☑ The write request is transmitted through read cache module, and then passed to the aSAN server directly, since it is a write request.
- ☑ The request is passed to the server and then data is written to HDD. After the write operation is complete, a result will be returned.
- ☑ The result is transmitted through the read cache module. If the result indicates the write operation is successful, it goes to Step 4; if it indicates the write operation is failed, the result will be directly returned to file access module without going to Step 4.
- ☑ This step consists of two other steps: 4.1 and 4.2. The data block [A1, A2] will be copied and saved to SSD by the read cache module, and a corresponding index will be created for this data block as well. This is Step 4.1. And the result of the write operation will be returned to file access module simultaneously. This is Step 4.2. Step 4.1 and 4.2 are executed at the same time. Therefore, the cache operation does not cause any delay to the write operation.

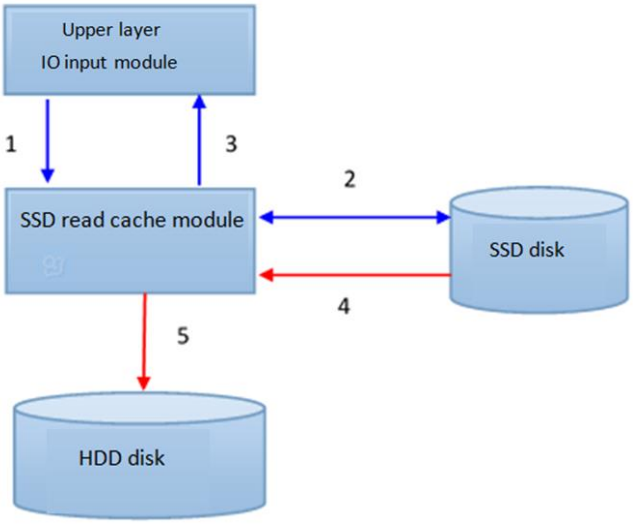
2.4.2.5 **SSD Write Caching**

SSD write caching operates on aSAN server and is supported since version aSAN2.0. Therefore, there is a write cache on each copy of data, i.e., SSD write cache has multiple copies as well. If an SSD fails, data security can be guaranteed with help of copies of write cache.



SDD Write Cache

SDD write caching is implemented by adding an SSD write cache on HDD, as shown in the diagram below:



Data stream flc ıto two types (blue and red). The blue data stream indicates a virtual machine is writing data to SSD write cache, while the red data stream indicates data is being read from SSD write cache and then written to HDD. The following are the illustration of the diagram above:

☑ SSD write cache module receives a write request from a virtual

machine.

- ☑ The write cache module writes data to SSD and obtains a value returned from SSD.
- ☑ If data is written to SSD successfully, the write cache module will send acknowledgement to that virtual machine that the data has been written successfully.
- ☑ When the data cached on SSD reaches a certain amount, part of data will be read from SSD and then written to HDD by the write cache module.

Step 4 and 5 run automatically in background and will not affect execution of Step 1, 2 and 3.

SSD Write Cache Hit

Writing data to HDD from SSD is triggered only when data on SSD reaches a certain amount. If there is a read request, SSD write cache will check whether the requested data is in the write cache. If the data is in the write cache (called a "cache hit"), it will be returned by SSD write cache; if it is not in the cache (called a "cache miss"), it will be returned from HDD.

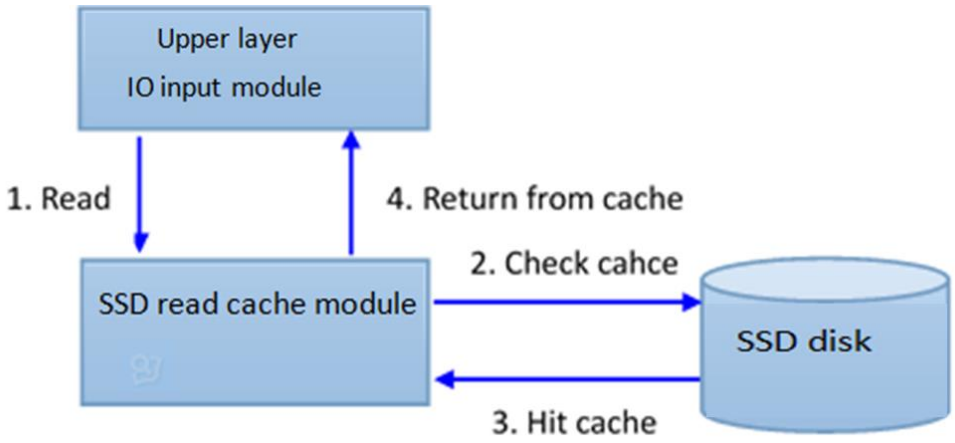


Illustration of the diagram above:

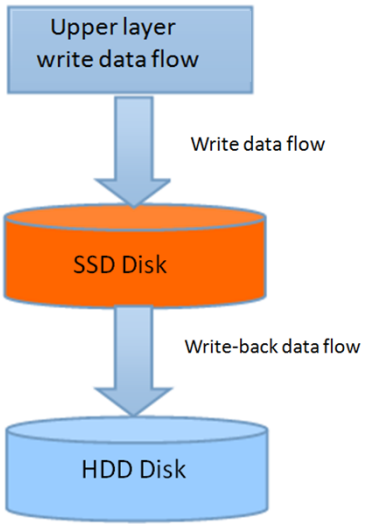
- ☑ A virtual machine initiates a read request.
- ☑ SSD write cache checks whether the requested data is in the

cache

- ☑ If cache hit occurs, the requested data will be returned by SSD write cache. Otherwise, cache miss occurs and requested data will be returned from HDD.
- ☑ Return the requested data to the virtual machine.

Handling of Full SSD Write Cache

If virtual machines write data to SSD continuously, SSD will get full and then the write speed of virtual machines may be as slow as the speed of data written to HDD from SSD.



If SSD is full, wri ır slower than the speed of data written to HDD from SSD. So is the situation with virtual machines. If such case often occurs, it indicates that SSD space is insufficient and needs to be increased to ensure I/O write performance.

Handling of Failed or Offline SSD

As mentioned before, SSD write cache operates on aSAN server and has multiple copies. When an SSD fails, data will not get lost as long as other SSDs on which copies of data are located operate properly. If an SSD has been offline for over 10 minutes, data on that SSD becomes invalid and then the copies of data on it will be repaired. That is to say, if SSD is unplugged by mistake, it must be plugged in 10 minutes, otherwise, all the data copies on it will be reconstructed.

2.4.3 aSAN Storage Data Reliability Safeguard

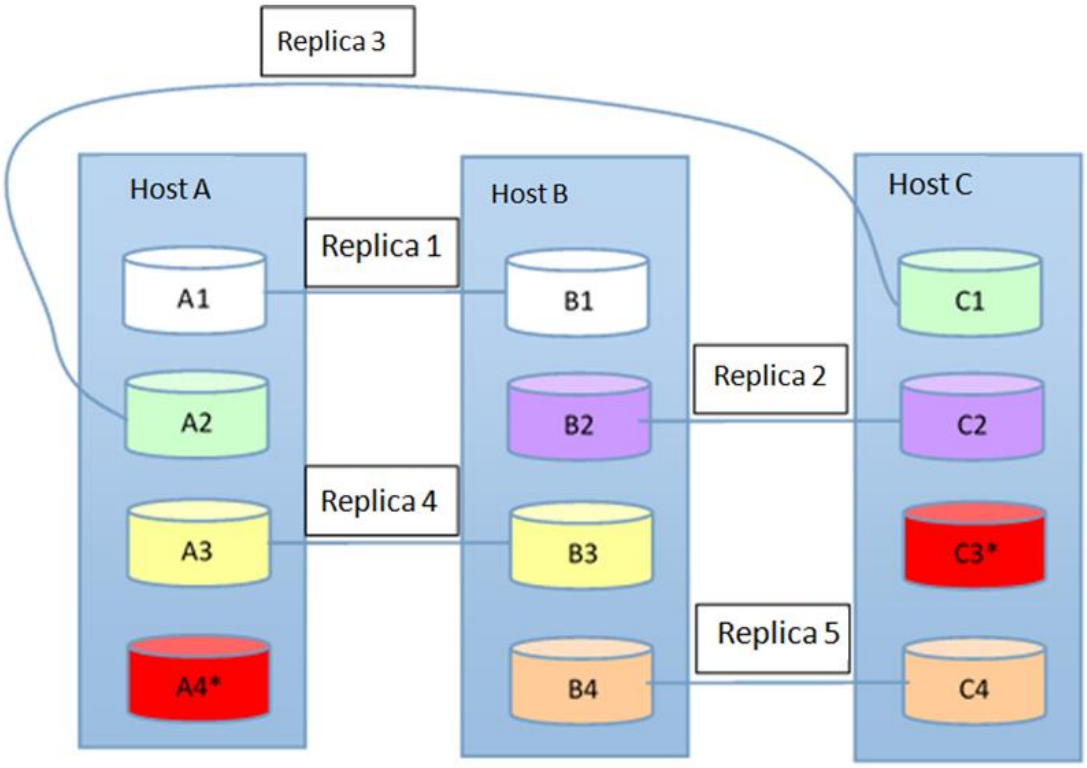
2.4.3.1 **Disk Failure Handling**

In case of disk failure, if network operator fails to fix the issue in specified period of time, aSAN will start to reconstruct the data on another disk, to ensure overall data integrity and reliability. That is also when spare disk is involved.

> Spare disks are disks defined automatically for global use in the cluster when aSAN initializes the storage. It is not necessary that every node is assigned a spare disk. Spare disk is an unused online disk and not in replicated volume. Thus, spare disk capacity will not be mapped to storage pool of aSAN.

For example, at least two spare disks will be reserved if there are two copies of data, while three spare disks will be reserved for three copies of data. Those spare disks are distributed on different nodes, rather than on one node.

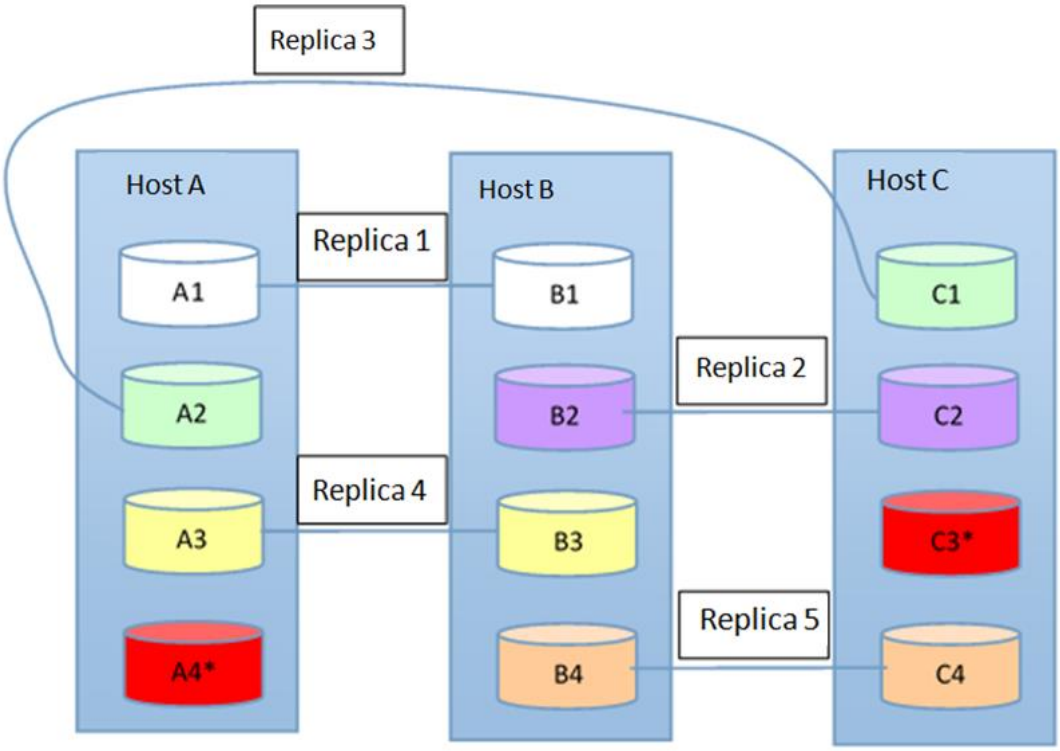
Take the following for example. Three nodes have 4 disks and store two data copies.



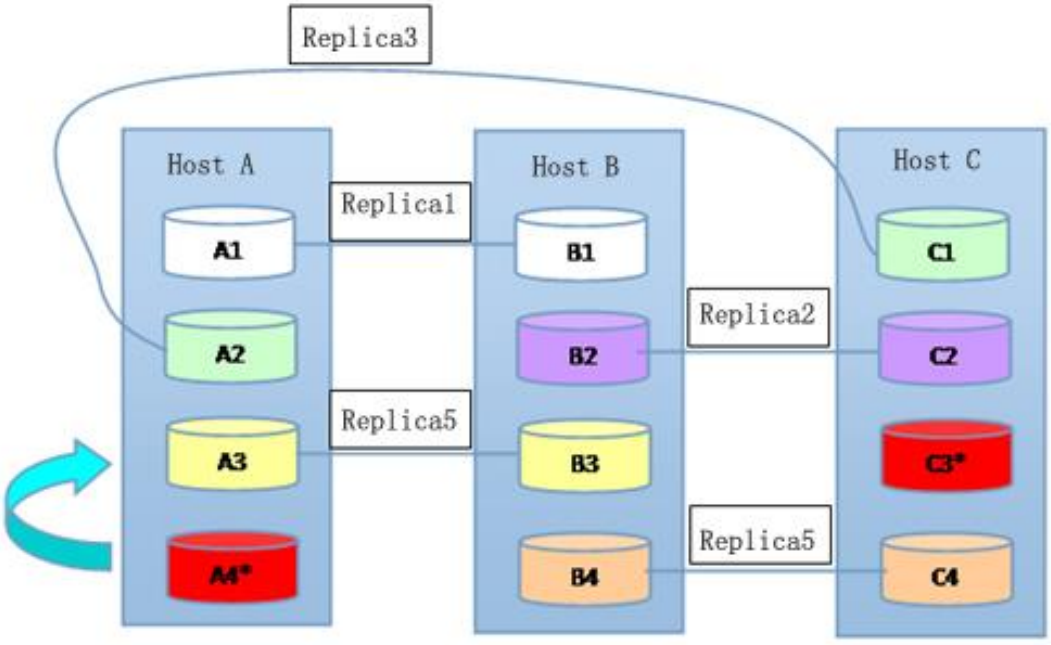
The above diagram shows how the disks on the three nodes are grouped. Disk A4 and C3 are reserved for spare disks and not joined to aSAN storage pool.

If any disk fails, failed disk will be automatically replaced with disk A4 or C3.

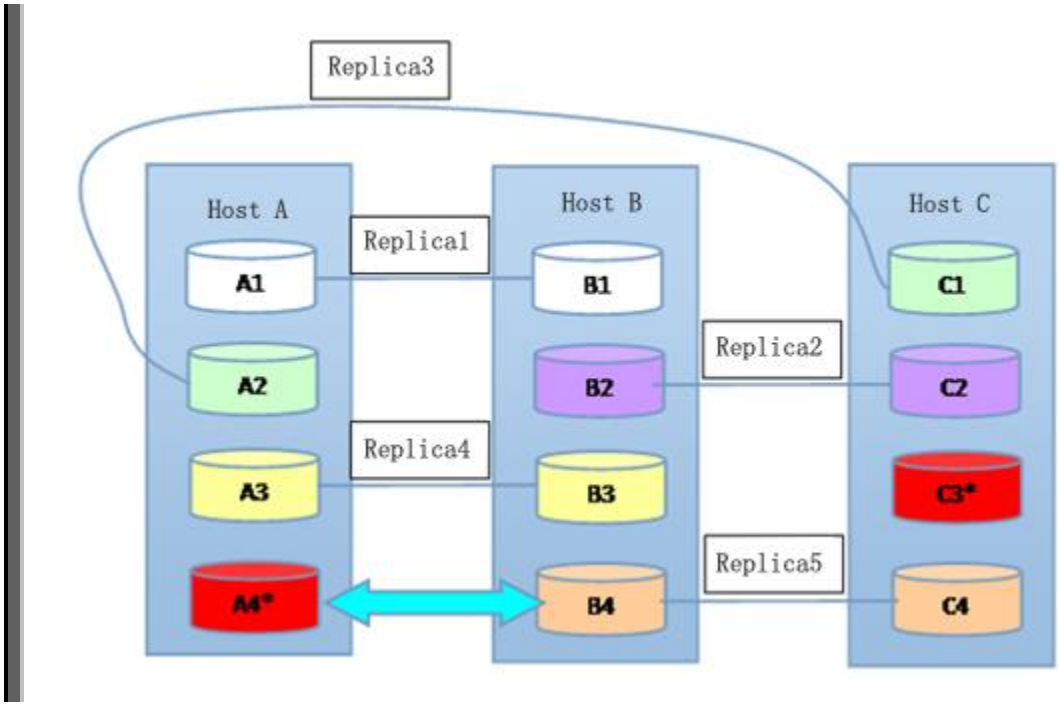
☑ Case 1: If disk C2 fails, it could be replaced with disk C3 or C4.



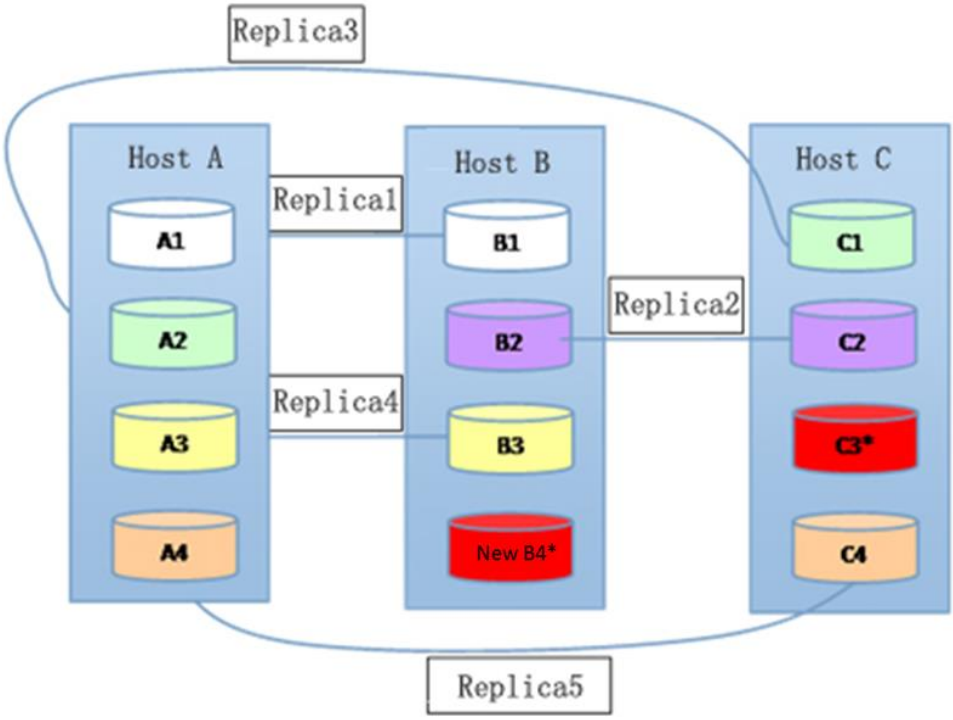
☑ Case 2: If disk A3 fails, it could be replaced with disk C3 or C4.



☑ Case 3: If disk B4 fails, it could only be replaced with disk A4, because disk C3 and C4 are on a same node.



On the GUI of Sangfor , it still displays information of the failed disk even it has been replaced with a spare disk. The faulty disk could be replaced with a new disk. If it is replaced with a new disk, that new disk will be used as a new spare disk. Thus, there is no need to move data to the new spare disk, which is different from moving data in the situation without use of space disk.



Take Case 3 for example. If disk B4 fails, disk B4 and C4 will be replaced with the spare disk A4 and then a new replicated volume 5 will be built. When failed disk is replaced with a new disk, that new disk B4 will be used as a new spare disk without moving data block.

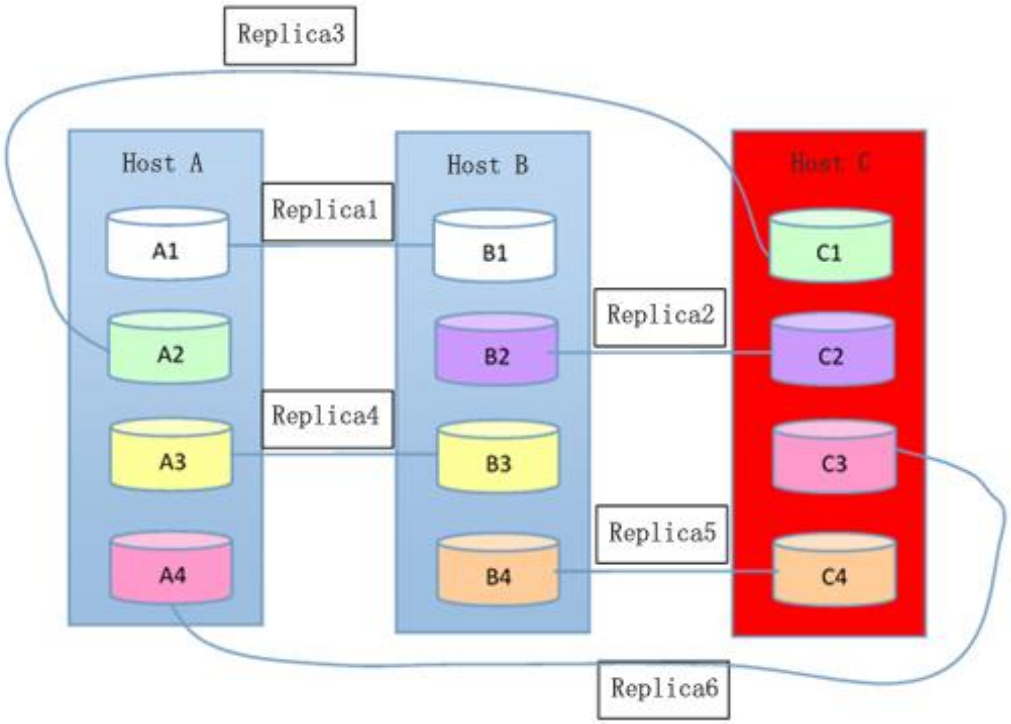
The process of failed disk replacement, including data reconstruction, can be done without interrupting services. That is much more reliable than RAID that requires services to stop.

2.4.3.2 **Node Failure Handling**

In the multi-node cluster, replicated volume is created across nodes in order to ensure data availability in case of node failure.

In the scenario that two nodes store two copies of data, data is still available when node B fails, because there is another copy of data on node A available.

The following is a relatively complex example without spare disk taken into account:

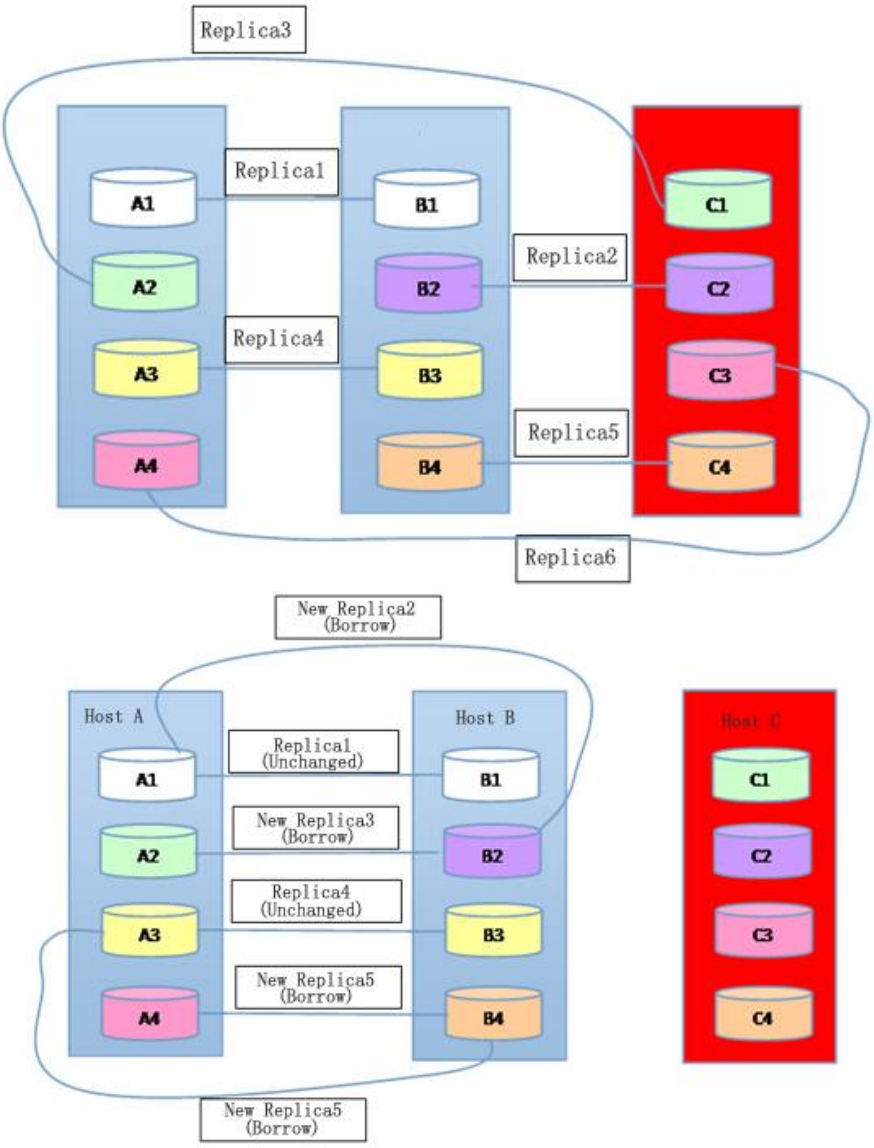


When node C is offline, data availability is not affected, because there is at least one copy of data available in other online replicated volumes.

Host data reconstruction is triggered after a specified period of time which could be set in Storage > Settings > Advanced > Host Data Reconstruction on web admin console of SSDC.

If node failure has not been handled manually within a specified period of time, host data will be automatically reconstructed on another node.

Assume that there are three nodes with two copies of data and one of the nodes, node C, fails, as shown in the diagram below:



By comparing the two diagrams, it is clear that new replicated volumes are automatically reconstructed by re-grouping disks on node A and B after node C fails for a specified period of time. Data reconstruction in case of node failure is essentially the same as that in case of disk failure, but the difference is that multiple copies of data will be reconstructed in the former scenario. If spare disk exists, it will be used instead.

Data reconstruction upon node failure will consume space of other disks and affect their performance. Therefore, the better way to solve node failure issue is to replace the failed node. Data reconstruction can be disabled in Storage > Settings > Advanced.

2.4.3.3 **Data Copy Rapid Restoration**

Copy of data sync means that a copy of data on a disk that gets online again after being offline will be synchronized from another copy of data on another online node so as to ensure that copies of data are consistent. A typical scenario is that copies of data become inconsistent when a node is disconnected for a while. In this case, the copy of data on that node should be synchronized from another copy of data on on another node.

With use of copy of data sync, only the different part of data, instead of the whole file, will be synchronized to the copy of data on a node that gets online again after being offline for a while. What's more, priority of IO for business is higher than IO for data sync, preventing the former from being affected by the latter.

2.4.4 aSAN Key Features

2.4.4.1 **Thin Provisioning**

Thin provisioning is an advanced, intelligent and high efficient virtualization technology to provision and manage storage capacity. It could provide a large capacity of virtual storage to operating system even the physical storage capacity is small. As the amount of business data increases, storage capacity could be expanded as required. In a word, thin provisioning only allocates the storage space needed, which can improve storage utilization dramatically.

For use of traditional way to provision storage capacity, users need to make a proper resource allocation plan in advance based on storage requirements of the current business and its development in the future. In operation, part of allocated storage capacity remains unused due to an incorrect estimate of business system scale. For example, 5TB capacity has been allocated to a business system, but only 1TB is used. In this case, the other 4GB is wasted and hard to be used by other business systems. Even for an excellent network administrator, it is hard to make a proper storage capacity allocation without any resource

waste. According to statistics from this industry, too large storage capacity will be allocated with use of pre-allocation, leading to about 30% of total storage capacity unused.

By using thin provisioning, aSAN solves storage capacity allocation problems and enhances resource utilization. What is thin provisioned is a virtual volume, rather than not a physical storage. The actual physical storage is only allocated based on a corresponding policy when data is written to the storage.

2.4.4.2 aSAN Private Network Link Aggregation

Link aggregation of aSAN is designed to enhance stability and performance of storage area network. It is implemented by the system and does not require to configure additional configuration on physical switches but proper connection.

For traditional link aggregation, link is assigned based on host IP address. Every two nodes are connected with one physical link. However, link is assigned based on TCP connection for aSAN link aggregation to achieve load balancing. We know that TCP connections between two physical nodes can be established on different physical links. Therefore, TCP-based link aggregation is more reliable and improves link utilization.

2.4.4.3 Data Copy Consistency

aSAN ensures data consistency using data consistency protocol, i.e., data is successfully written into disk only when it is written to all the copies of data. In general cases, copies of data are identical. If a disk fails, data will not be written to the copy of data on that disk but synchronized to it after the disk is recovered. If the disk fails for a long time or forever, it will be removed from the cluster and a new disk will be found to store that copy of data which is reconstructed using data reconstruction mechanism.

2.5 aNET (Network)

2.5.1 aNET Overview

Network virtualization is also a very important part in constructing HCI. Therefore problems may occur if we still use traditional IT infrastructure:

- ☑ Migrating virtual machines without changing network policies is a problem.
- ☑ Virtualized report center covers a lot of services. As for cloud service, traditional VLAN is far from meeting requirements for service segregation and service continuity, therefore, safe micro-segmentation on a massive scale for different users and services is a big issue to be solved. Service continuity with IP and security policies following the migration of VMs is highly in demand
- ☑ Deployment, flexibility, scalability and cost of network should be improved for construction and deployment of service system on virtualized report center.
- ☑ In traditional network, both basic IT infrastructure and applications are controlled by specific devices, which are expensive, less capable and less flexible, and moreover unable to satisfy the need for fast, flexible and automatic network configuration.

To address the above problems, Sangfor provides Overlay + NFV solution, also called aNET. With vxLAN Overlay, large L2 network be easily achieved and users are safely segregated at bridge level. With NFV, all sorts of network resources (including basic routing and switching, security, application delivery, etc.) are allocated as required and are scheduled flexibly. By combining vxLAN Overlay and NFV,

network virtualization is successfully realized.

2.5.2 aNET Working Principle

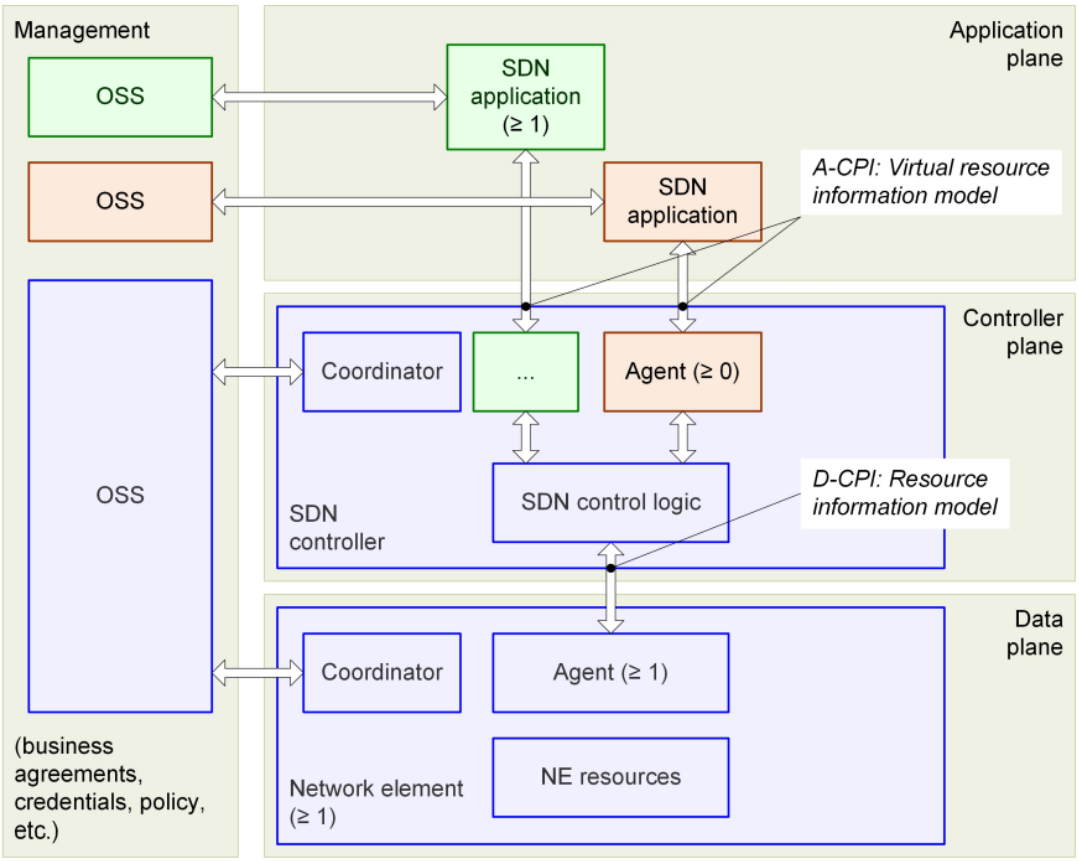
2.5.2.1 **SDN**

SDN (Software Defined Network) is a creative network infrastructure. It utilizes standardized technologies (e.g., OpenFlow) to separate controller plane from data plane, and makes network traffic management more flexible, centralized and granular, paving the way for central management and application optimization. Furthermore, it will make network more programmable, flexible, scalable, automated and controllable than ever before, to easily adapt to the changing business demands.

From the perspective of implementation of SDN, SDN can be viewed both from a broad sense and a narrow sense.

- ☑ SDN from a broad sense: It mainly includes Network Virtualization (NV), here mainly refers to Overlay, and Network Function Virtualization (NFV).
- ☑ SDN from a narrow sense: It mainly refers to implementation of SDN through OpenFlow.

A standard SDN diagram is as follows:



Sangfor aNET has SDN controller integrated into HCI management GUI and on top of that we use innovative "what you draw is what you get" provisioning white board to automate network design and provision.

The Data Plane sits in the kernel to achieve high performance.

In terms of the routing and switching features, Sangfor HCI aNET provides basic L2 switching for vSwitch and static routing for vRouter for the moment and it is more than enough for most of our enterprise customers. In the future, we will extend the support of Open flow and integrate according to the latest open flow standard. The SDN controller will remain part of Sangfor HCI management platform.

2.5.2.2 **NFV**

Open but not isolated, generalized but not specialized, NFV (Network Function Virtualization) is what that can extract and virtualize conventional network functions from specialized network appliances,

and make them run on generic hardware platforms. NFV (Network Function Virtualization) aims at using widely used hardware to carry all sorts of software, so that software can be loaded flexibly and can be quickly configured at such places as report center, network nodes and clients, therefore, network deployment and adjustment speed will be enhanced, deployment complexity and overall investment costs will be reduced, and network devices are unified, generalized and more adaptive.

NFV and SDN are complementary to each other in that NFV makes deployment more flexible, and SDN adds more flexibility and convenience.

With NFV, network resources are virtualized and become flow resources. Overlay model helps movable network resources to get rid of physical boundaries, and makes it possible for those resources to flow as required just like a resource pool, therefore, ultimately, network resources in HCI are defined flexibly, allocated as required and adjusted when needed.

☑ aNET Kernel Implementation: High-performance platform

In traditional virtualization environment, data packets are received by network interface cards (NICs), and then are classified, and specified actions are generated and executed upon data packets. In traditional Linux mode, it takes a large amount of the total time for the system to receive and send data packets, i.e., it takes a lot of time to send data packets from incoming port to outgoing port even though user space does nothing.

When NIC receives data frame, it will send data frame to pre-provisioned kernel buffer via DMA (Direct Memory Access), update one appropriate descriptor ring, and then send out interruption to notify the arrival of data frame. The operating system handles interruption, updates the descriptor ring, and sends data frame to

network stack. To send data packets to local socket, data will be duplicated into the local socket and then user space which has that socket will receive data.

User space writes data into socket with system call, and kernel of Linux will copy data from user buffer to kernel buffer. Then network stack processes data, encapsulates it as required, and uses a NIC driver. The NIC driver will update one appropriate descriptor ring and will notify NIC that there is a transmission task to be carried out.

The NIC transfers data frame from kernel buffer to built-in FIFO buffer, and later data frame will be sent to the Internet. Then the NIC will send an interruption to notify that data frame has been successfully sent, so that the kernel will release buffer related to the data frame.

In traditional mode, CPU consumption mainly occurs in the following situations:

- ☑ **Interruption**: It refers to suspend the current task when receiving interruption, and schedule soft IRQ program to execute tasks scheduled by interruption. With the increase of traffic load, it will take more and more time for the system to handle interruption, and performance will be greatly affected by speed of NIC when traffic speed reaches 10Gbps. But when there are multiple NICs with speed reaching 10Gbps, system will be flooded by interruption, and all the services will be greatly affected.
- ☑ **Context Switch**: It refers to save register and status information of the current thread, and later recover register and status information of the preempted thread, so that the thread will start again from where it has been interrupted. Both scheduling and interruption will bring about context switch.
- ☑ **System Call**: It will switch user mode to kernel mode, and then back to user mode, which will clear data in pipe and affect

caching.

☑ **Data Copying**: Data frame will be copied from kernel buffer to socket, and then again be copied from socket to kernel buffer. How long the process takes is subject to data volume to be copied.

☑ **Scheduling**: Scheduling program makes each thread run for a short period of time, resulting in a false impression that multiple tasks are executed concurrently in the kernel. When scheduling timer interrupts or is at other time, Linux scheduling program will start and check whether the current thread has expired. When the scheduling program is about to run another thread, then context switch will occur.

2.5.2.3 Implementation of aNET

Data Plane

On generic OSes like Linux, network applications and non-network applications are fairly treated. For that reason, I/O throughput will never be set too high in design phase. In data plane design, Sangfor aNET learned from netmap and dpdk solutions, specifically for IO intensive network applications.

☑ Special NICs and generic NICs are supported

Programmable NICs such as e1000e, igb, ixgbe, bnx2, tg3 and bnx2x of Intel and Broadcom support high performance schemes, whereas NICs such as e1000 support generic schemes, so that hardware compatibility is ensured.

☑ Global memory pool is shared across kernels and processes

With global memory system for multiple cores and processes, data packets received by NIC are copied once for all without being copied again while data is transmitted to kernel, application layer and virtual machines. Memory pool automatically increases and unused memory will also be automatically reclaimed.

☑ Interruption and context switch prevention

Single threads are locked to hardware threads, thus, context switch,

thread switch, and interruption between kernel and users are eliminated. Meanwhile, each thread has caching; therefore, there is no need to compete for buffer.

Ideally, information that is required to process data packets should be in caching of the kernel before data packets arriving at the system. Just imagine, if lookup table, context of data flow, and connection control block are already in cache before data packets arrive, then data packets can be directly processed without being mounted or waiting for the completion of sequential memory access.

☑ Data plane of application layer is more stable

A small bug in kernel state may bring about system crash, as for the application layer, the worst situation is that process is terminated. But we have inspection and monitoring system, with which, network will recover in seconds without being felt by virtual machines even in worst cases.

Data plane, core of the whole system, is responsible for forwarding messages, and is composed of multiple data transmission threads and one control thread. Data transmission thread is responsible for processing messages, and control thread is responsible for receiving messages from control processes.

In data thread, fast path and slow path are separated to process messages. Messages are forwarded based on session. One data flow matches one session. The first message of the data flow is responsible for searching for all sorts of table entries, creating sessions, and recording results of table entries into session. The subsequent messages of the data flow only need to search for sessions, process and forward messages according to what has been recorded in sessions.

All the messages in the system are received by data thread. As for messages to be forwarded there is no need for them to be sent to Linux protocol stack, but are directly processed in data thread and sent from NIC. As for messages forwarded to devices (for example, SSH, TELNET, OSPF, BGP, DHCP, etc.), they cannot be directly processed by data thread, therefore, those messages should be sent to Linux protocol stack through TUN interface, and should go through data thread before

being sent out.

When using Longest prefix matching (LPM) on 6-core, 2.0 GHz, Intel Xeon processors L5638, as for four cores of the total 6 cores, each core has one thread and four 10G Ethernet ports, in this situation, IP layer forwarding performance of the 64-byte data packets reaches 9 million pps, which is almost 9 times of that of original Linux (double processors, 6 cores, 2.4GHz, forwarding performance is 1 million pps).

Data plane enables kernel or service layer to interact with data packet IO, and application protocol stack provides optimized network stack. Compared with Linux SMP, it is less dependent on kernel of Linux, thus is much more expandable and reliable.

Control Plane

With the aid of data plane and protocol stack, control plane accomplishes a great many functions. Such functions are available as DHCP, RSTP and DNS proxy. Those services are directly available to virtual machines without the need of installing similar third-party software.

2.5.3 aNET Network Functions

2.5.3.1 aSW (Virtual Switch)

aSW is used to manage virtual switches on multiple nodes, including management of physical ports on nodes and virtual ports on virtual machines.

By logically combining switches of multiple nodes in the cluster into a large centralized switch, aSW saves the effort of configuring every switch one by one, and provides centralized control for network connection. Therefore, deployment, management and monitoring of virtual network connection are simplified, and thus is applicable to

large-scale network deployment.

aSW guarantees that network configuration of virtual machines are the same while being migrated among different nodes, meanwhile, it provides a great many network configuration management functions, such as dynamic port binding, static binding, IP access control, virtual machine QoS, so that unified management on network resources and real-time network monitoring are achieved.

2.5.3.2 **aRouter (Virtual Router)**

Router is an inevitable component in SDN. aNET provides virtualize router to external network, as well as other functions, such as VLAN port, NAT rule, ACL policy, DHCP address pool, DNS proxy, etc.

Besides, router provided by aNET has function of HA which is the same as that of virtual machines. Router will automatically migrate to normal nodes so that failure will be recovered quickly. Therefore, reliability of service network in HCI is assured, and service downtime is reduced.

2.5.3.3 **vAF**

Security is of paramount importance in constructing network virtualization. vAF (Application Firewall) provides overall protection from L2 to L7, efficiently recognizes risks both from network layer and application layer by detecting flow bidirectionally, and offers much more powerful protection against attacks from application layer than deploying multiple security devices at the same time such as traditional firewalls, IPS, WAF.

Moreover, users can generate service-based risk reports to get insight of security status of network and service system, and to improve management efficiency and operational cost eventually.

2.5.3.4 **vAD**

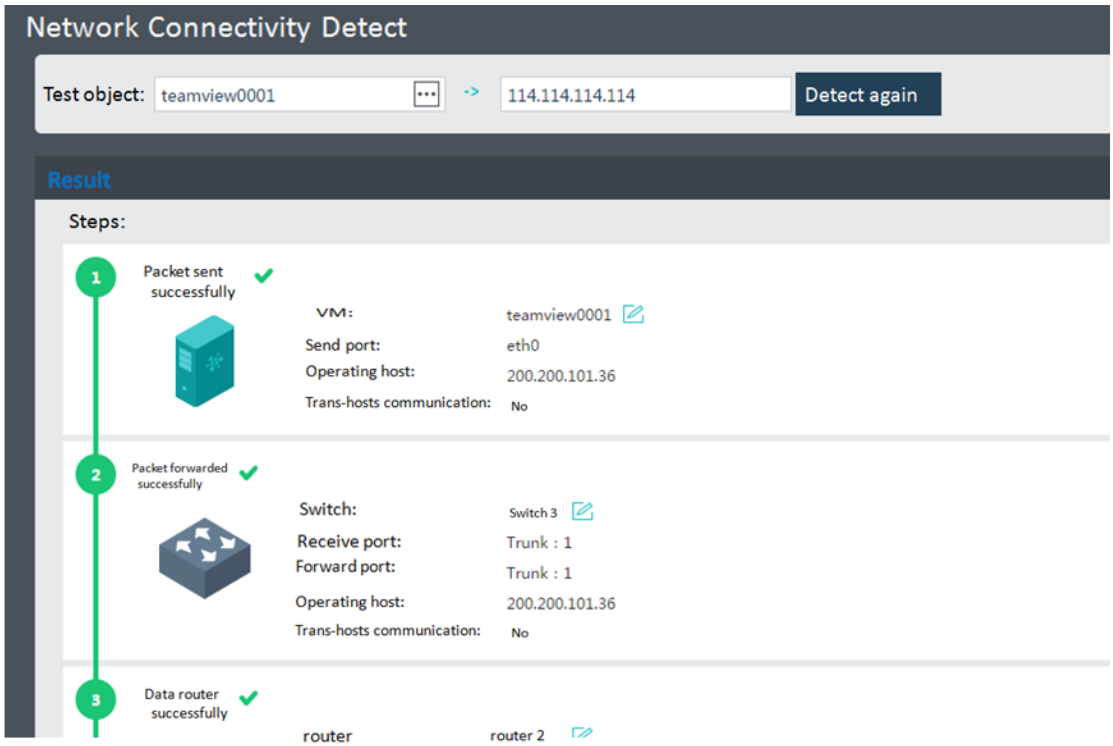
vAD is the upgraded version of traditional load balancing controller,

which integrates servers, links, data center load balancing and application performance optimization, therefore, it is a best choice for constructing reliable and efficient service infrastructure. User can improve reliability of application system, links and data center, and improve bandwidth utilization, server performance and user experience.

2.5.4 aNET Special Technology Features

2.5.4.1 Network Connection Detection

Network detecting is to send tagged ICMP message, trace and record every hop of the message on the forwarding plane, including name of virtual devices and forwarding ports, and then forwarding path of message is made clear by combining all the information that has been noted down. It is just like detecting with Tracert, but is much more detailed, because it can detect outgoing path and incoming path as well for us to quickly find network problems.



2.5.4.2 Traffic Visualization

Data packets are noted down by every port of any virtual device when being forwarded through forwarding plane. By searching for entries kept by ports of virtual devices, uses can get big picture of the network traffic.

2.5.4.3 **"What You Draw is What You Get" Topology**

Logic typology featuring what you see is what you get is a special function of Sangfor HCI. Since aSAN can be accessed through aSV, together with network resource pool provided by aNET, all sorts of resources are available for constructing different logic typologies. When logic typologies are being constructed, kernel or service layer of HCI will execute a great many commands, and will simulate environment of the kernel or service layer based on logic typologies, so that it is easy for IT managers to quickly draw a typology for software-defined data center.

2.5.4.4 **Distributed Firewall**

Today 80% of data traffic is inside datacenter; only 20% is south-north traffic. Traditional firewall is typically deployed at the perimeter of the network; it can't effectively prevent attacks that take place inside data center. Modern attacks exploit inherent weaknesses in traditional perimeter-centric network security strategies to infiltrate enterprise data centers. Non-core business with low level security protection can be breached easily by hackers and be exploited as a springboard to compromise other more important businesses. With distributed firewall, it's like putting a firewall on the exit and entrance of each VM. Once the policy is configured, the backend resource will be adjusted dynamically to protect the business at any time no matter whatever change is on the topology, VM location or IP. Sangfor distributed virtual firewall has the ability to receive changes of user configuration, topology and IP, and then dynamically update them to the firewall through self-developed network control plane.

3. Introduction to Sangfor

3.1 Product Offering

Currently, Sangfor HCI could be purchased with appliance or pure software

- ☑ Applicance named aServer: This is a full solution integrated with all the HCI software and sangfor server hardware.
- ☑ Pure Sangfor HCI with 3 packages:
  - Sangfor HCI Basic license which includes aCenter,aSV and basic aNET
  - Sangfor HCI Premium license which includes aCenter,aSV，basic aNET and aSAN
  - Sangfor HCI Enterprise license which includes aCenter,aSV，aNET, aSAN and NFV capability

Sangfor HCI uses x86 servers, which should meet the following minimum configuration requirements:

- ☑ CPU: Intel CPU supporting VT
- ☑ Memory: >=8 GB
- ☑ Hard Disk: SSD of data center, for example, intel S3500, S3510, S3700, P3500, P3700, Samsung 840 DC, SanDisk DC, Kingston DC SATA 7200, SAS 10k, SAS 15k
- ☑ RAID: JBOD mode (transparent mode)
- ☑ NIC: 8 GE or 10 GE.

4. Core Values of Sangfor

4.1 Reliability

With HCI, Sangfor HCI integrates computing, storage, network and security into a large resource pool; therefore, it can quickly construct service systems and provides reliability by employing multiple techniques without damaging service performance, for example, hot migration, HA, data backup and recovery, multiple copies.

Up until June 2017, there are over 50K CPU processors running on Sangfor HCI. It has been proven as a mature and reliable technology since years back.

4.2 Security

Sangfor HCI platform is secured with built Firewall and WAF module to protect any attack towards the platform itself.

Sangfor HCI platform provides VM-VM micro-segmenation

Sangfor HCI provides data encryption at VM level.

Sangfor HCI provides 3 levels of platform administration privileges.

Sangfor HCI provides integrated NGFW, Antivirus, WAF and real security visibility scanner and visualized report.

4.3 Easy to use

Unlike those traditional hardware defined IT architecture, Sangfor innovatively evolves the IT architecture with both hardware consolidation of all network equipment as well as compute and storage. From management perspective, Sangfor HCI allows IT administrator to manage all technologies through one management interface with ease.

There is no need to have deep understanding of each technology anymore. IT admin could simply map a wanted IT topology on to the HCI management interface, all the provisioning work could be done automatically.

4.4 TCO reduction

CAPEX Savings	**How much cost does sangfor HCI reduce**
Hardware savings	30% -60%
Software license cost	30% -70%
Overprovisioned hardware	30% -80%
Management product	30% -60%
OPEX Savings	How much cost does sangfor HCI reduce
Human resource	40% -80%
Space	40% -70%
Electricity	40% -60%

4.5 TTM and Agility

Instead of having 3-4 specialist working together for months to build up a data center, with sangfor HCI, one junior IT admin is able to build up a complete data center within days.