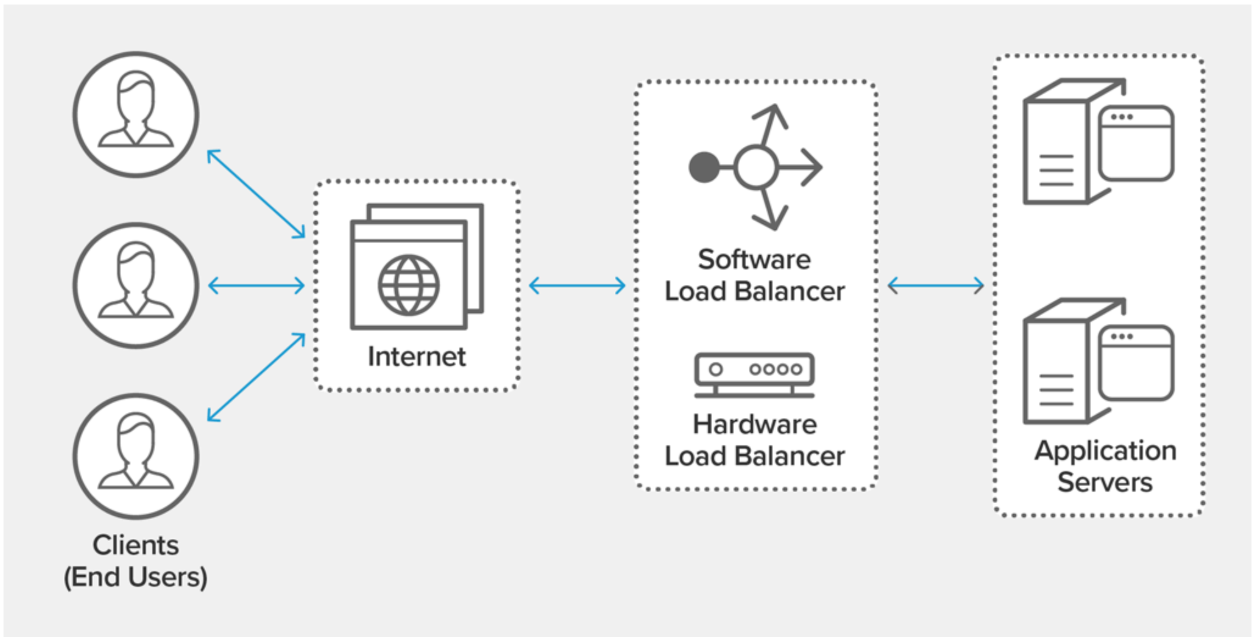
UNIT IV
Ch 9: Dynamic Load Balancing

Load Balancing (LB)

The load balancing problem concerns both system administrators and application developers. Because it affects both the efficiency of the system and the performance of applications. Specially in HPC the amount of resources that are wasted with a bad load distribution can be extremely large.

A load balancer acts as the “traffic cop” sitting in front of your servers and routing client requests across all servers capable of fulfilling those requests in a manner that maximizes speed and capacity utilization and ensures that no one server is overworked, which could degrade performance. If a single server goes down, the load balancer redirects traffic to the remaining online servers. When a new server is added to the server group, the load balancer automatically starts to send requests to it.
In this manner, a load balancer performs the following functions:

- Distributes client requests or network load efficiently across multiple servers
- Ensures high availability and reliability by sending requests only to servers that are online
- Provides the flexibility to add or subtract servers as demand dictates



Introduction to Dynamic Load Balancing

Dynamic load balancing is a technique used in computer systems to distribute workloads across multiple resources (such as servers, CPUs, or network links) in real-time. The goal of dynamic load balancing is to optimize system performance by ensuring that each resource is utilized efficiently and that no single resource becomes a bottleneck. There are several algorithms that can be used for dynamic load balancing, such as round-robin, least connections, and IP hash. Additionally, many load balancing solutions also include features such as health checking and automatic failover to ensure high availability.

Dynamic load balancing is commonly used in distributed systems, cloud computing, and web-based applications to distribute incoming requests across multiple servers or resources. It allows for efficient use of resources and can help prevent system failures caused by overloading a single resource.

Load Balancing Algorithms

Load balancing uses various algorithms. Different load balancing algorithm gives different benefits. Use of anyone of these varies on application owners need. Some examples are following -

- **Round Robin Method:** This method rotates servers by sending traffic to the first available server and then moves that server to the bottom of the queue.
- **Least Connection Method:** It selects the server with the fewest active connections. This method works well when there are a large number of permanent connections in the traffic unevenly distributed between the servers.
- **The Least Response Time Method:** This method selects the service with the fewest active connections and the lowest average response time.
- **The Least Bandwidth Method:** This method directs traffic to the server that is currently serving the least amount of traffic, measured in megabits per second (Mbps).
- **IP Hash:** the IP address of the client determines which server receives the latest incoming request.

Application Life Cycle with Dynamic Load Balancing

The application life cycle with dynamic load balancing would include the typical stages of an application life cycle with additional considerations related to load balancing.

• Planning: In this stage, the project team would identify the business need and objectives for the application, and develop a plan for how the application will meet those needs. This includes determining the requirements for the application, as well as any constraints that may impact the development process. The team would also consider the load balancing requirements of the application, such as the expected number of users, traffic patterns, and resource utilization.

• Design: In this stage, the project team creates detailed designs for the application, including user interface, data model, and system architecture. The team would also design the load balancing architecture and choose the appropriate load balancing algorithm and software.

• Development: In this stage, the project team writes the code for the application, based on the designs created in the previous stage. The team would also implement the load balancing functionality, test it and ensure that it works as expected.

• Deployment: In this stage, the application is deployed to a production environment where it will be used by end-users. This stage also includes configuring the application for the production environment, as well as any necessary training or documentation for end-users. In addition, the load balancer and its configuration are deployed to the production environment as well.

• Maintenance: Once the application is deployed, it enters the maintenance stage. This includes fixing any bugs that are discovered, updating the application to meet new requirements, and making any necessary performance improvements. The load balancer is also continuously monitored and adjusted as needed to ensure optimal performance.

• Retirement: Eventually, the application will reach the end of its useful life and will need to be retired. This stage includes an assessment of the application's usefulness, ensuring that data from the application is preserved, and planning for the replacement of the application. The load balancer would also be retired or replaced as necessary.

Overall, the application life cycle with dynamic load balancing would involve additional considerations related to load balancing, such as designing and implementing the load balancing functionality, monitoring, and adjusting it to ensure optimal performance.

Who can use DLB?

Any application written in C, C++ or Fortran in any of the supported parallel programming models. The current supported parallel programming models are the following:

MPI+OpenMP

MPI+OmpSs

OmpSs (Multiple Applications)

We are open to adding support for more programming models in both inner and outer level of parallelism.

How does DLB work?

DLB will use the malleability of the inner level of parallelism to change the number of threads of the different processes running on the same node. There are different load balancing algorithms implemented within DLB. They all rely on this main idea but they target different types of applications or situations.

In fig. 4 we can see an example of a DLB load balancing algorithm. In this case, the application is running two MPI processes on a computing node, with two OpenMP threads each one. When MPI process 1 reaches a blocking MPI call it will lend its assigned CPUs (number 1 and 2) to the second MPI process running in the same node. This will allow MPI process 2 to finish its computation faster.

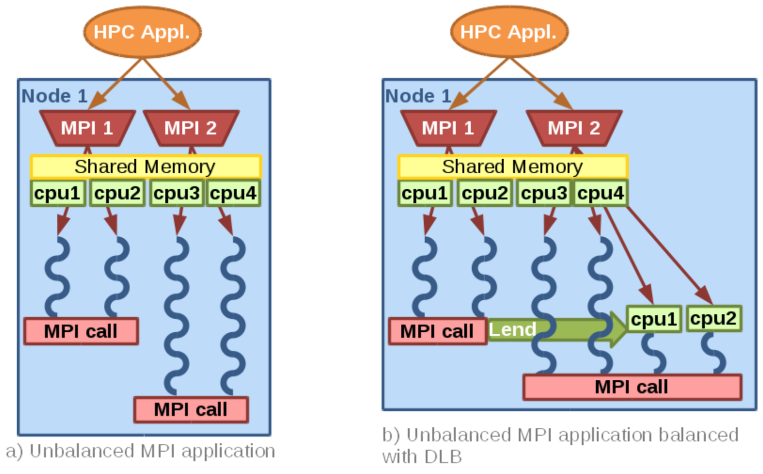
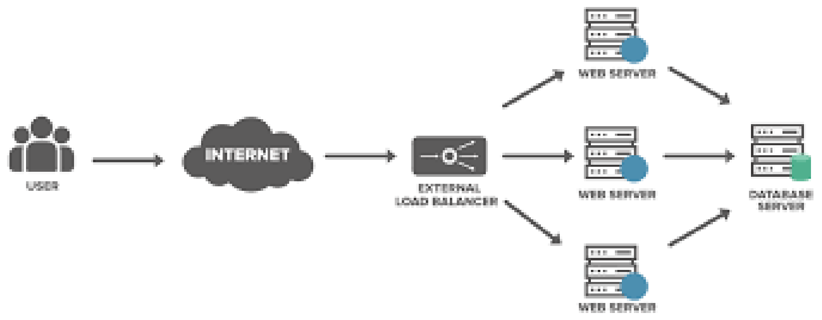


Fig 4: Example of DLB behavior

Use of Dynamic Load Balancing

Dynamic load balancing is used in a variety of contexts to distribute workloads across multiple resources and improve system performance. Some of the common use cases for dynamic load balancing include:

• Web applications: Dynamic load balancing is commonly used in web-based applications to distribute incoming requests across multiple servers. This can help ensure that the application remains responsive and available, even during periods of high traffic.



• Cloud computing: In cloud computing environments, dynamic load balancing is used to distribute workloads across multiple virtual machines, ensuring that resources are used efficiently and that no single machine becomes a bottleneck.
• Distributed systems: In distributed systems, dynamic load balancing is used to distribute tasks across multiple machines, ensuring that the system as a whole can handle the workload efficiently.
• Networking: Dynamic load balancing can also be used in networking environments to distribute traffic across multiple links, ensuring that bandwidth is used efficiently and that no single link becomes a bottleneck.
• Gaming: In gaming environments, dynamic load balancing can be used to distribute players across multiple game servers, ensuring that resources are used efficiently and that no single server becomes overwhelmed.
• Big Data: Dynamic load balancing can also be used to distribute large data sets across multiple machines, ensuring that the data can be processed efficiently and that no single machine becomes a bottleneck.

Working of Dynamic Load Balancing

Dynamic load balancing works by distributing incoming requests or workloads across multiple resources in real-time. The load balancer, which is a separate component of the system, receives incoming requests and then directs them to the appropriate resource based on a set of predefined rules or algorithms.

The process can be broken down into a few key steps:
• Request routing: The load balancer receives incoming requests and routes them to the appropriate resource based on the chosen algorithm. This can include determining which resource currently has the least number of active connections, or using a round-robin approach to distribute requests evenly across all resources.
• Resource monitoring: The load balancer continuously monitors the resources to determine their current status and capacity. This can include monitoring resource utilization, such as CPU or memory usage, as well as checking the health of each resource.

• Load balancing algorithm: The load balancer uses a load balancing algorithm to decide which resource should handle the incoming request. The algorithm takes into account the current status of the resources, the specific characteristics of the incoming request and the predefined rules.
• Request forwarding: Once the load balancer has determined which resource should handle the incoming request, it forwards the request to that resource. The resource then processes the request and sends the response back to the client.
• Automatic failover: If a resource becomes unavailable or fails, the load balancer can automatically redirect requests to a different resource, ensuring that the system remains available even in the event of a failure.
The process of dynamic load balancing is continuous and happens in real-time, ensuring that resources are used efficiently and that no single resource becomes a bottleneck. It allows for the system to handle a large number of requests and maintain high availability.

Ch 10: Parallel meshing and Remeshing

Getting Started with Parallel Meshing and Remeshing

A parallel meshing technique using a combination between a local remeshing technique and repartitioning is presented. The meshing method is based on local mesh topology optimizations and is shown to work for all meshing applications from adaptive remeshing to mesh generation by using a minimal volume principle. Parallel remeshing is performed independently on each subdomain with fixed interfaces. A constrained repartitioning technique is introduced to move the interfaces between subdomains in an optimal way. Repartitioning and remeshing are iterated until a good mesh and a good partition are reached. Several examples are given for different meshing objectives. Application examples are shown with the commercial code Forge3, devoted to large deformation analysis

Large Deformation and Adaptive Remeshing

Large deformation and adaptive remeshing are techniques used to handle large deformations in a mesh, such as those that occur in simulations of mechanical processes like impact, crash, and explosion. In large deformation problems, the mesh can become highly distorted or even collapse, making it difficult to obtain accurate results. Adaptive remeshing is a way to overcome these difficulties by automatically refining or coarsening the mesh based on the level of deformation.

The process of adaptive remeshing typically involves the following steps:

• Initial mesh generation: A coarse mesh is generated that covers the entire domain of interest.

• Deformation simulation: A simulation of the deformation is run using the initial mesh, and the deformed shape is computed.

• Error estimation: The error in the solution is estimated by comparing the computed deformed shape with the true deformed shape.

• Adaptive mesh refinement: Based on the error estimate, the mesh is refined or coarsened in areas where the deformation is large or where the error is high.

• Repeat steps 2-4: The simulation is run again using the refined mesh, and the process is repeated until the error is within a specified tolerance.

The meshing approach used in leads to a complete solution both for mesh generation and for adaptive meshing and remeshing. However, this technique was first developed to perform remeshing when large deformations are involved in material forming process simulations [4], [5]. In case of Lagrangian simulation of the forging process, the mesh gets distorted as the part deforms and, without remeshing, can rapidly degenerate. Meshing and adaptive remeshing are often a key point in such applications. The remeshing method is mainly based on simple local improvement mechanisms. However, it is also a mesh generation method. The algorithm enables the mesh topologies to be modified with few geometric constraints. A simple local process makes possible to iterate from one mesh to a new one by improving the mesh topology and by adding or deleting nodes until a suitable mesh is obtained. Moreover, changes of the surface mesh and the volume mesh can be strongly coupled. Eventually, adaptive meshing with respect to a mesh size map is just a local mesh optimization. Parallel remeshing in the context of this approach was already experienced in [7] and is more deeply addressed in this paper.

Partitioning and parallel meshing technique

The parallel unstructured mesh generation strategies differ by the treatment of the interface between subdomains or/and by the domain decomposition (the partitioning) which can be mesh based or model based. In the first case, a mesh must exist initially and is decomposed by using a mesh based partitioner to provide subdomains to be remeshed. In the second case, the geometric model itself is decomposed using a geometric partitioner and a mesh generator can run on each subdomain.

Partitioning and parallel meshing are techniques used to divide a large mesh into smaller sub-meshes, which can then be processed in parallel on multiple processors or cores. The process of partitioning a mesh involves dividing the mesh into smaller sub-meshes, known as partitions, such that each partition can be processed independently on a separate processor or core.

There are several different partitioning algorithms available, each with their own advantages and disadvantages. Some popular partitioning algorithms include:

• Metis: A popular open-source partitioning library that uses a multilevel graph partitioning algorithm

• ParMETIS: A parallel version of Metis that is designed for use on distributed memory systems

• Scotch: An open-source partitioning library that uses a multilevel nested dissection algorithm.

There are several different algorithms and techniques used for parallel mesh generation, including:

• Domain Decomposition: This method divides the computational domain into smaller sub-domains, which are then meshed separately and combined to form the final mesh.

• Space-Filling Curves: This method uses space-filling curves, such as the Hilbert curve or the Morton curve, to divide the computational domain into smaller sub-domains.

• Load Balancing: This method uses a load balancing algorithm to distribute the mesh generation process across multiple processors, ensuring that each processor is working on a similar amount of data.

• Metis: This is a popular library for graph partitioning, and it can be used to partition the mesh into smaller sub-meshes that can be distributed across multiple processors.

• Distributed Data Structures: This method uses distributed data structures, such as distributed octrees or distributed hash tables, to store and distribute the mesh data across multiple processors.

Parallel mesh generation can be done in a variety of software packages and tools, such as the DistMesh, Metis, and the ParMETIS library, among others. Its important to note that the choice of parallel mesh generation algorithm and software package will depend on the specific requirements of the simulation or analysis task, as well as the available computational resources.

The remeshing is performed independently on each subdomain without changing the mesh at the interfaces between subdomains, as shown in the forging example of Fig. 2.

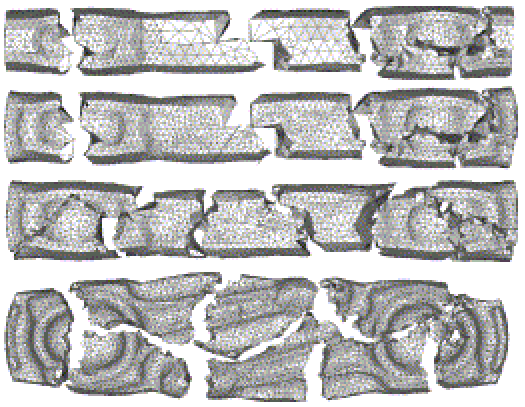


Fig. 2. Parallel remeshing for large deformation by combining local remeshing an repartitioning.

Parallel mesh generation

Parallel mesh generation in numerical analysis is a new research area between the boundaries of two scientific computing disciplines: computational geometry and parallel computing.[1] Parallel mesh generation methods decompose the original mesh generation problem into smaller sub problems which are solved (meshed) in parallel using multiple processors or threads. The existing parallel mesh generation methods can be classified in terms of two basic attributes:

1. the sequential technique used for meshing the individual subproblems and
2. the degree of coupling between the subproblems.

While many solvers have been ported to parallel machines, grid generators have left behind. Still the preprocessing step of mesh generation remains a sequential bottleneck in the simulation cycle. That is why the need for developing of stable 3D parallel grid generator is well-justified.
A parallel version of the MeshSim mesh generator by Simmetrix Inc.,[10] is available for both research and commercial use. It includes parallel implementations of surface, volume and boundary layer mesh generation as well as parallel mesh adaptivity. The algorithms it uses are based on those in reference [4] and are scalable (both in the parallel sense and in the sense that they give speedup compared to the serial implementation) and stable. For multicore or multiprocessor systems, there is also a multithreaded version of these algorithms that are available in the base MeshSim product.

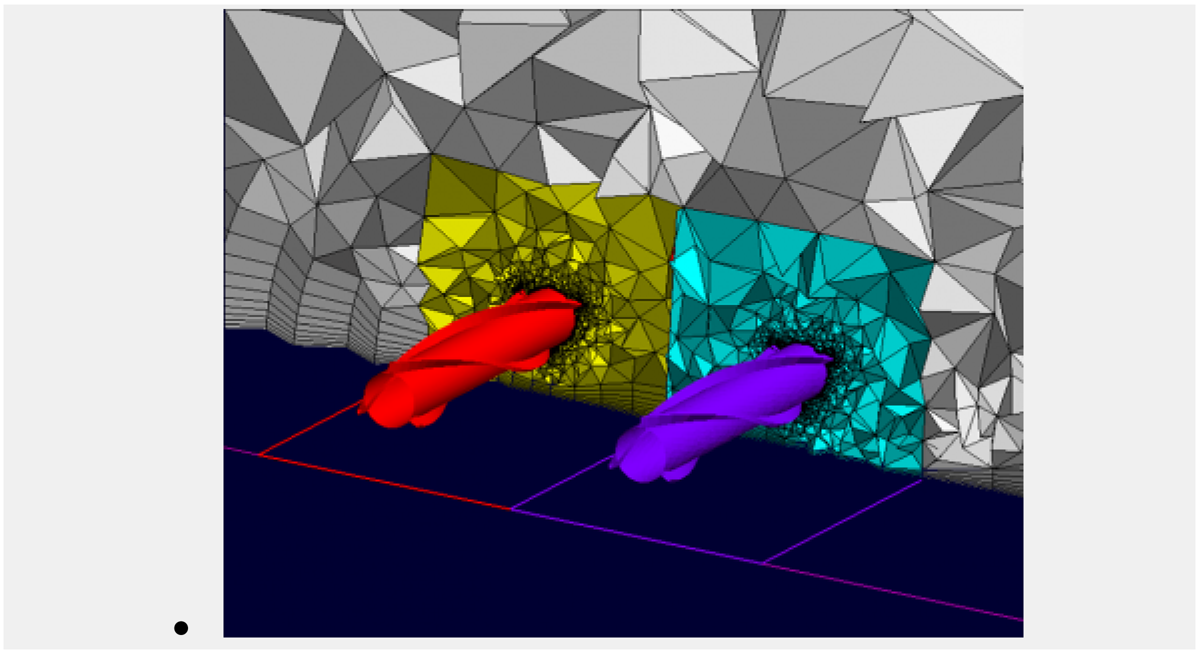
Parallel (Distributed) Mesh Generation

CENTAUR has the capability to generate hybrid meshes in parallel. The domain is distributed to separate processors and/or machines allowing for simultaneous meshing within all subdomains.

- CAD-based surfaces used to divide the domain (planar, cylindrical, etc).
  - Linux and Windows machines can be combined
    - Straightforward setup
  - Multiple processors can be used with a single key.

Example: Two Straked Pipes

This geometry is split into three zones to allow for each of the two pipes to be replaced using modular mesh generation. The mesh for each zone is generated simultaneously using the CENTAUR's parallel mesh generation feature.



Ch 11: Network and Storage options for Advanced Computing.

What is the Network Storages?

The Network storages are defined as a special type of dedicated data storage server that includes storage devices such as disk arrays, CD / DVD drives, tape drives or removable storage media, and embedded system software that provide cross-platform file sharing capabilities.

The Network storages usually occupy their own node on a LAN without the intervention of an application server, allowing users to access data on the network. In this configuration, network storage centrally manages and processes all data on the network, loads the data from applications or Uninstall the enterprise server, effectively reduce the total cost of ownership and protect the user's investment.

Types of Network Storages

For the enterprise storage devices, network storages are mainly divided into three types: DAS, SAN, and NAS, according to their implementation modes. They provide different solutions for different application environments respectively.

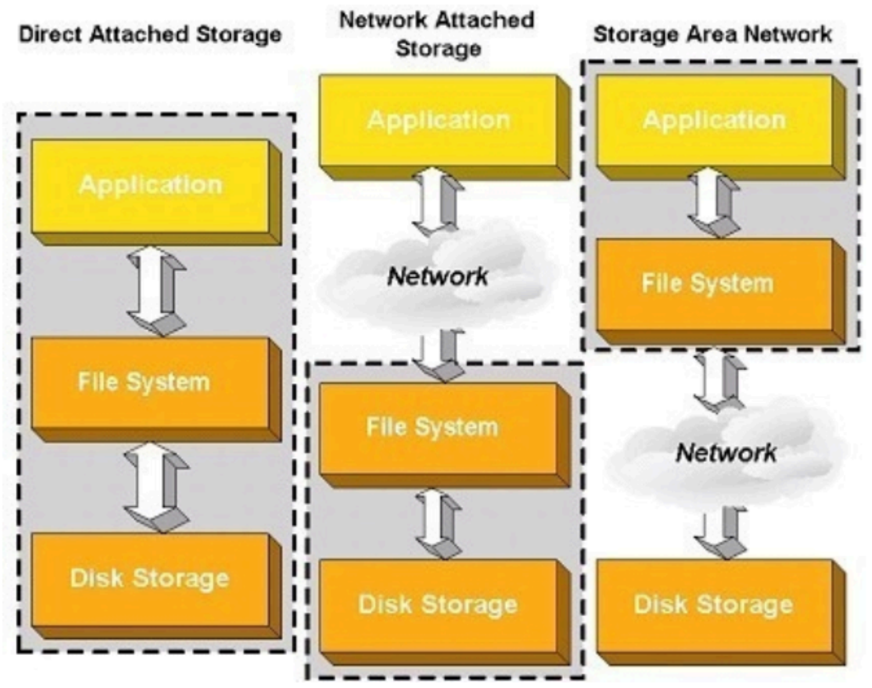


Figure 1. Differences among three types of Network Storages.

DAS

DAS (Direct Attach Storage) is a storage type that attach the host server directly. Every host server has the independent storage device and every host server can't attach other host server. When they need to send and receive data, it needs a complicated configuration. If each host server run a different operating system, the rule will be more complicated.

Usually, the DAS is ideal for the simple network. It don't need to save too much data. This type of network storage is a elder technology.

SAN

SAN（Storage Area Network）is a storage type which is used high-speed (optical fiber) network to attach the host server. This SAN will deploy at the end of those host servers. It used I/O connecting types, such as SCSI, ESCON and Fiber-Channels.

SNA is designed for those application environments that need high-speed network, secure data, good function of data sharing, such as telecommunications, banking, the amount of large data applications. Its features are high-cost and good quality.

NAS

NAS (Network Attached Storage), a network storage device, which is usually directly connected to the network and provides data access service. A NAS storage device acts as a data file service system and is characterized by its high cost performance. It's ideal for education area, government, business and other data storage applications.

What is a hard disk drive?

A computer hard disk drive (HDD) is a non-volatile data storage device. Non-volatile refers to storage devices that maintain stored data when turned off. All computers need a storage device, and HDDs are just one example of a type of storage device.

HDDs are usually installed inside desktop computers, mobile devices, consumer electronics and enterprise storage arrays in data centers. They can store operating systems, software programs and other files using magnetic disks.

More specifically, hard disk drives control the reading and writing of the hard disk that provides data storage. HDDs are used either as the primary or secondary storage device in a computer. They are commonly found in the drive bay and are connected to the motherboard via an Advanced Technology Attachment (ATA), Serial ATA, parallel ATA or Small Computer System Interface (SCSI) cable, among other formats. The HDD is also connected to a power supply unit and can keep stored data while powered down.

A hard disk drive -- often shortened to *hard drive* -- and hard disk are not the same things, but they are packaged as a unit and either term can refer to the whole unit.

Why do computers need hard disks?

Storage devices like hard disks are needed to install operating systems, programs and additional storage devices, and to save documents. Without devices like HDDs that can retain data after they have been turned off, computer users would not be able to store programs or save files or documents to their computers. This is why every computer needs at least one storage device to permanently hold data as long as it is needed.

How do hard disk drives work?

Most basic hard drives consist of several disk platters -- a circular disk made of either aluminum, glass or ceramic -- that are positioned around a spindle inside a sealed chamber. The platter spins with a motor that is connected to the spindle. The chamber also includes the read/write heads that magnetically record information to and from tracks on the platters using a magnetic head. The disks also have a thin magnetic coating on them.

The motor spins the platters at up to 15,000 rotations per minute. As the platters spin, a second motor controls the position of the read and write heads that magnetically record and read information on each platter.

Hard disk drive storage capacity

Some of the most common storage drive capacities include the following:

- **16 GB, 32 GB and 64 GB.** This range is among the lowest for HDD storage space and is typically found in older and smaller devices.
- **120 GB and 256 GB.** This range is generally considered an entry point for HDD devices such as laptops or computers.
- **500 GB, 1 TB and 2 TB.** Around 500 GB and above of HDD storage is typically considered decent for an average user. Users can most likely store all their music, photos, videos and other files with this much space. Individuals with games that take up a lot of space should find 1 TB to 2 TB of HDD space suitable.
- **More than 2 TB.** Anything over 2 TB of HDD space is suitable for users who work with high-resolution files, who need to store or house a large amount of data, or who want to use that space for backup and redundancy.

Currently, the highest capacity HDD is 20 TB. However, an HDD actually has less space than advertised, as the operating system, file system structures and some data redundancy procedures use a portion of that space.

Hard drive components and form factors

Hard disk drive components include the spindle, disk platter, actuator, actuator arm and read/write head. Even though the term can refer to the unit as a whole, the term *hard disk* is the set of stacked disks -- in other words, the part of the HDD that stores and provides access to data on an electromagnetically charged surface.

The HDD form factor refers to the physical size or geometry of the data storage device. HDD form factors follow a set of industry standards that govern their length, width and height, as well as the position and orientation of the host interface connector. Having an industry-standard form factor helps determine a common compatibility with different computing devices.

The most common form factors for HDDs in enterprise systems are 2.5-inch and 3.5-inch -- also known as small form factor (SFF) and large form factor (LFF). The 2.5-inch and 3.5-inch measurements represent the approximate diameter of the platter within the drive enclosures.

While there are other form factors, by 2009, manufacturers discontinued the development of products with 1.3-inch, 1-inch and 0.85-inch form factors. The falling price of flash made these other form factors almost obsolete. It is also important to note that while nominal sizes are in inches, actual dimensions are specified in millimeters.

Many solid-state drives (SSDs) are also designed for the HDD form factor. SSDs that fit into the same slots as HDDs generally use the SATA or serial-attached SCSI (SAS) interface to transfer data to and from the host computing system.

What are external HDDs?

Most HDDs are found internally in a computer and work as stated above. However, individuals can also purchase external hard drives. External hard drives can be used to expand the storage capacity of a computer or to act as a portable device to back up data. External drives connect to a computer or device through interfaces like USB 2.0, USB-C or with External SATA (eSATA). External hard drives may also have slower data transfer rates compared to internal HDDs.

The main advantage of an external hard drive, aside from being able to expand a device's storage space, includes being portable. Users can store data from multiple devices and physically bring that data with them wherever they go.

Common hard disk errors

Hard disks can fail for all sorts of reasons. However, failures generally fall into the following six broad categories.

- **Electrical failure** occurs when, for example, a power surge damages a hard disk's electronic circuitry, causing the read/write head or circuit board to fail. If a hard disk powers on but cannot read and write data or boot, it is likely that one or more of its components has suffered an electrical failure.
- **Mechanical failure** can be caused by wear and tear, as well as by a hard impact, like a hard drop. This may cause, among other things, the read/write drive head to hit a rotating platter, causing irreversible physical damage.

- **Logical failure** results when the hard disk's software is compromised or ceases to run properly. All sorts of data corruption can lead to a logical failure. This includes corrupted files, malware and viruses, improperly closing an application or shutting down a computer, human error or accidentally deleting files that are critical to hard disk functionality.
- **Bad sector failure** can occur when the magnetic media on a hard disk's rotating platter is misaligned, resulting in a specific area on the platter becoming inaccessible. Bad sectors are common and often limited when they occur. Over time, however, the number of bad sectors can increase, eventually leading to a system crash, inaccessible files or the hanging or lagging of the operation of a hard disk.
- **Firmware failure** happens when the software that performs the maintenance tasks on a drive and enables the hard disk to communicate with a computer becomes corrupted or stops working properly. This type of failure can lead to the disk freezing during bootup or the computer a hard disk is connected to not recognizing or misidentifying it.
- **Multiple unknown failures** that accumulate over time can also occur. For example, an electrical problem could lead to a mechanical failure, such as a read/write head crash. It might also lead to a logical failure, resulting in several bad sectors developing on the hard disk platters.

Understanding the NAND Flash Drive

What is NAND flash memory?

NAND flash memory is a type of non-volatile storage technology that does not require power to retain data. An important goal of NAND flash development has been to reduce the cost per bit and to increase maximum chip capacity so that flash memory can compete with magnetic storage devices, such as hard disks. NAND flash has found a market in devices to which large files are frequently uploaded and replaced. MP3 players, digital cameras and USB flash drives use NAND technology.

NAND flash saves data as blocks and relies on electric circuits to store data. When power is detached from NAND flash memory, a metal-oxide semiconductor will provide an extra charge to the memory cell, keeping the data. The metal-oxide semiconductor typically used is a floating-gate transistor (FGT). The FGTs are structured similar to NAND logic gates.

NAND memory cells are made with two types of gates, control and floating gates. Both gates will help control the flow of data. To program one cell, a voltage charge is sent to the control gate.

NAND flash memory operation

Flash memory is a special type of electronically erasable programmable read-only memory (EEPROM) chip. The flash circuit creates a grid of columns and rows. Each intersection of the grid holds two transistors separated by a thin oxide layer -- one transistor is called a *floating gate* and the other is called the *control gate*. The control gate connects the floating gate to its respective row in the grid.

History and evolution of NAND flash memory

Flash memory traces its roots to the development of metal-oxide-semiconductor field-effect transistors (MOSFETs). MOSFET technology was developed in 1959, with the development of floating gate MOSFETs coming in 1967. Developers of these early transistors realized that the devices could hold states without external power and proposed their use as floating gate memory cells for programmable read-only memory (PROM) chips that would be both non-volatile and reprogrammable -- a potential boon in flexibility over existing ROM chips. These transistors formed the foundation of erasable PROM (EPROM) and EEPROM devices through the 1970s, though their use was limited.

Types of NAND flash storage

Common types of NAND flash storage include SLC, MLC, TLC, QLC and 3D NAND. What separates each type is the number of bits per cell. The more bits stored in each cell, the less expensive the NAND flash storage would cost.

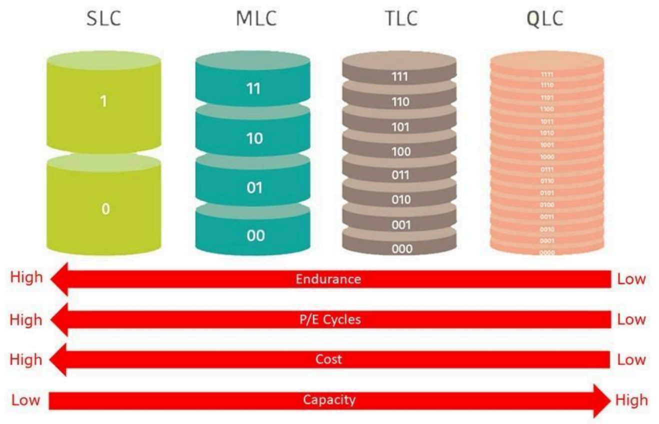
SLC, or single-level cells, store one bit per cell. SLC has the highest endurance but is also the most expensive type of NAND flash storage.

MLC, or multi-level cells, stores two bits per cell. Because erasures and write cycles occur two times more, MLC has less endurance compared to SLC. However, it's less expensive. Many PCs will use MLC.

TLC, or triple-level cells, store three bits per cell. Many consumer-level products will use this because it is less expensive, however lower-performing.

QLC, or quad-level cells, store four bits per cell. QLCs have even less endurance and are generally less expensive.

3D NAND -- 2D or planar NAND has only one layer of memory cells, whereas 3D NAND stacks cells on top of one another. Samsung refers to 3D NAND as Vertical NAND or V-NAND.



Data Center Storage Configurations

In this article in the series, Robert Sheldon explains the differences between direct-attached storage, network-attached storage, and storage area networks.

Direct-Attached Storage

As the name suggests, DAS is a storage configuration in which HDDs or SSDs are attached directly to a computer, rather than connecting via a network such as Ethernet, Fibre Channel, or InfiniBand. DAS typically refers to HDDs or SSDs. Other storage types, such as optical or tape drives, can theoretically be considered DAS if they connect directly to the computer, but references to DAS nearly always refer to HDDs or SSDs, including those in this article.
DAS can connect to a computer internally or externally. External DAS can be a single drive or part of an array or RAID configuration. Whether internal or external, the DAS device is dedicated to and controlled by the host computer.

A computer's DAS drive can be shared so that other systems can access the drive across the network. Even in this case, however, the computer connected to the drive still controls that drive. Other systems cannot connect to the drive directly but must communicate with the host computer to access the stored data.

DAS connects to a computer via an interface such as Serial-Attached SCSI (SAS), Serial Advanced Technology Attachment (SATA), Small Computer System Interface (SCSI), or Peripheral Component Internet Express (PCIe). Along with other storage technologies, the interface can have a significant impact on drive performance and is an important consideration when choosing a DAS drive. (See the first article in this series for information about interfaces and related storage technologies.)

Some IT teams turn to DAS because it typically provides better performance than networked storage solutions such as NAS and SAN. When using DAS, the host server does not need to contend with potential network bottlenecks such as sluggish network speed or network congestion, and the data is by definition in close proximity to the server. Other systems that connect to the host might run into network issues, but the host itself—and the applications that run on it—have unencumbered access to data hosted on DAS.

DAS is also cheaper and easier to implement and maintain than networked systems such as NAS or SAN. A DAS device can often be implemented through a simple plug-and-play operation, with little administrative overhead. Because DAS storage includes a minimal number of components, other than the SSD or HDD itself, the price tag tends to be much lower than the networked alternatives.

DAS is not without its downsides, however. Because a server can support only a relatively small number of expansion slots or external ports, DAS has limited scalability. In addition, limitations in the server's compute resources can also impact performance when sharing a drive, as can the data center's network if contention issues arise. DAS also lacks the type of advanced management and backup features provided by other systems.

Despite these disadvantages, DAS can still play a vital role in some circumstances. For example, high-performing applications or virtualized environments can benefit from DAS because it's generally the highest performance option, and DAS eliminates potential network bottlenecks. In addition, small-to-medium sized businesses—or departments within larger organizations—might turn to DAS because it's relatively simple to implement and manage and costs less.

DAS can also be used in hyperscale systems such as Apache Hadoop or Apache Kafka to support large, data-intensive workloads that can be scaled out across a network of distributed computers. More recently, DAS has been gaining traction in HCI appliances, which are made up of multiple server nodes that include both compute and storage resources. The usable storage in each node is combined into a logical storage pool for supporting demanding workloads such as virtual desktop infrastructures (VDIs).

Network-Attached Storage

NAS is a file-level storage device that enables multiple users and applications to access data from a centralized system via the network. With NAS, users have a single access point that is scalable, relatively easy to set up, and cheaper than options such as SAN. NAS also includes built-in fault tolerance, management capabilities, and security protections, and it can support features such as replication and data deduplication.

A NAS device is an independent node on the local area network (LAN) with its own IP address. It is essentially a server that contains multiple HDDs or SSDs, along with processor and memory resources. The device typically runs a lightweight operating system (OS) that manages data storage and file sharing, although in some cases it might run a full OS such as Windows Server or Linux.

Users and applications connect to a NAS device over a TCP/IP network. To facilitate data transport, NAS also employs a file transfer protocol. Some of the more common protocols are Network File System (NFS), Common Internet File System (CIFS), and Server Message Block (SMB). However, a NAS device might also support Internetwork Packet Exchange (IPX), NetBIOS Extended User Interface (NetBEUI), Apple Filing Protocol (AFP), Gigabit Ethernet (GigE), or one of several others. Most NAS devices support multiple protocols.

NAS devices are generally easy to deploy and operate and relatively inexpensive when compared to SANs. In addition, users and applications running on the same network can easily access their files, without the limitations they might encounter if retrieving data from DAS. NAS devices can also be scaled out or integrated with cloud services. In addition, they provide built-in redundancy while offering a great deal of flexibility.

That said, a NAS device must compete with other traffic on the network, so contention can be an issue, especially if network bandwidth is limited. It should be noted, however, that NAS is often configured on private networks, which can help mitigate contention issues. However, too many users can impact storage performance, not only on the network, but also in the NAS device itself. Many NAS devices use HDDs, rather than SSDs, increasing the risk of I/O contention as more users try to access storage.

Because of the network and concurrency issues, NAS is often best suited for small-to-medium sized businesses or small departments within larger organizations. NAS might be used for distributing email, collaborating on spreadsheets, or streaming media files. NAS can also be used for network printing, private clouds, disaster recovery, backups, file archives, or any other use cases that can work within NAS's limitations, without overwhelming the network or file system.

When deciding whether to implement a NAS device, you should consider the number of users, types of applications, available network bandwidth, and any other factors specific to your environment. DAS might be the optimal choice because it's typically more performant, cheaper, and easier to set up than NAS. On the other hand, you might consider looking to a SAN for increased scalability and additional management features.

Storage Area Network

A SAN is a dedicated, high-speed network that interconnects one or more storage systems and presents them as a pool of block-level storage resources. In addition to the storage arrays themselves, a SAN includes multiple application servers for managing data access, storage management software that runs on those servers, host bus adapters (HBAs) to connect to the dedicated network, and the physical components that make up that network's infrastructure, which include high-speed cabling and special switches for routing traffic.

SAN storage arrays can be made up of HDDs or SSDs or a combination of both in hybrid configurations. A SAN might also include one or more tape drives or optical drives. The management software consolidates the different storage devices into a unified resource pool, which enables each server to access the devices as though they were directly connected to that server. Each server also interfaces with the main LAN so client systems and applications can access the storage.

There is a widespread myth that SANs are high-performing systems, but historically this has rarely been true. In fact, slow-performing SANs are ubiquitous across data centers and are first and foremost optimized for data management, not performance. However, now that SSDs are becoming more common, hybrid or all-flash SANs are bringing performance to the forefront.

Integral to an effective SAN solution is a reliable, high-performing network capable of meeting workload demands. For this reason, many modern SANs are based on Fibre Channel, a technology for building network topologies that can deliver high bandwidth and exceptional throughput, with speeds up to 128 gigabits (16 GB) per second. Unfortunately, Fibre Channel is also known for being complex and pricey, causing some organizations to turn to alternatives such as Internet SCSI (iSCSI), Fibre Channel over Ethernet (FCoE), or even NVMe over Fabrics (NVMe-oF).

With the right network topology and internal configuration in place, a SAN can deliver a block-level storage solution that offers high availability and scalability, possibly even high performance. A SAN includes centralized management, failover protection, and disaster recovery, and it can improve storage resource utilization. Because a SAN runs on a dedicated network, the LAN doesn't have to absorb the SAN-related traffic, eliminating potential contention.

However, a SAN is a complex environment that can be difficult to deploy and maintain, often requiring professionals with specialized skill sets. This alone is enough to drive up costs, but the SAN components themselves can also be pricey. An IT team might try to reduce costs by cutting back in such areas as Fibre Channel or licensed management capabilities, but the result could be lower performance or more application maintenance.

For many organizations—typically larger enterprises—the costs and complexities are worth the investment, especially when dealing with numerous or massive datasets and applications that support a large number of users. SANs can benefit use cases such as email programs, media libraries, database management systems, or distributed applications that require centralized storage and management.

Organizations looking for networked storage solutions often weigh SAN against NAS, taking into account complexity, reliability, performance, management features, and overall cost. NAS is certainly cheaper and easier to deploy and maintain, but it's not nearly as scalable or fast. For example, a NAS uses file storage, and a SAN uses block storage, which incurs less overhead, although it's not as easy to work with. Your individual circumstances will determine which storage system is the best fit. (For information about the differences between block and file storage, refer to the first article in this series).

Modern Storage Technologies

In this article of the series, Robert Sheldon discuses emerging trends in storage like virtual SANs, intelligent storage, computational storage and storage-class memory.

Software-Defined Storage

Although traditional storage configurations still play a vital role, they were not designed to meet the demands of today's massive amounts of dynamic, distributed, and heterogenous data. Some IT teams are addressing these challenges by turning to software-defined storage (SDS), a software-based solution that provides an abstraction layer between the applications and storage devices, in effect unbundling the storage software from the underlying hardware.

Ideally, an SDS solution will run on commodity servers and support a wide range of storage devices, removing any dependencies on proprietary hardware or its software. The SDS solution controls storage requests from the application, while managing the storage resources themselves. Separating the data plane from the control plane in this way can lead to greater operational agility and control over where and how data is stored.

Although vendors take different approaches to SDS, solutions commonly use virtualization to consolidate the physical storage devices into logical resource pools that can be dynamically

controlled and allocated to the applications that need them. An SDS solution exposes standards-based APIs for provisioning and managing resources, making it easier to automate operations, support development efforts such as infrastructure-as-code (IaC), and integrate with container orchestration tools such as Kubernetes.
One of the biggest advantages of SDS is flexibility. Not only do IT teams have more hardware choices, but applications also benefit because storage resources can be allocated and scaled on demand. In addition, an SDS solution can better utilize the physical resources, which can translate to lower costs, especially when you eliminate proprietary storage systems and the vendor lock-in that comes with them. SDS can sometimes even improve performance through the use of parallelism, data tiering, and data caching.
Yet SDS is not without its challenges. For starters, implementing and maintaining an SDS solution can be a complex undertaking, especially when working with multiple storage products from different vendors. These multi-vendor scenarios can also make it more difficult to get vendor support or even identify the source of a specific problem (further exacerbating support issues). In addition, SDS solutions might not be as hardware-agnostic as sometimes suggested, and some SDS products might not include all the features available to dedicated proprietary systems, although that has been steadily improving.

Virtual SAN

Another technology that organizations have been leveraging to help address modern workloads is the virtual storage area network (VSAN), a mechanism for separating and isolating traffic on networks such as Fibre Channel or Ethernet. In a VSAN configuration, the physical SAN is broken into logical partitions that segregate devices connected to the same fabric. For example, you can create VSANs to separate teams with different security or performance requirements, or you can use them to isolate backup traffic from production traffic.

This type of VSAN is different from what you see in products that use the term virtual SAN, VSAN, or even vSAN (lowercase “v”) to describe SDS capabilities. For example, VMware offers a product called vSAN (formerly Virtual SAN), an SDS solution used in conjunction with VMware vSphere to provide a foundation for hyperconverged infrastructures. VMware vSAN creates logical resource pools made up of the DAS devices connected to the infrastructure’s vSphere clusters, and then makes those resources available to the cluster’s virtual machines.

The VSAN in the context of this article has its roots in Cisco Systems and is specific to SAN implementations. Each logical VSAN supports the same operations and configurations available to the physical SAN, but they can be configured independently to meet specific needs. Devices within the vSAN can freely communicate with one another but cannot communicate with devices outside of their own VSAN, even if they’re connected to the same physical SAN. In this way, an organization can build a single SAN topology, but still have the benefits of logical topologies that are indifferent to the geographic locations of the SAN's switches and attached devices.

Intelligent Storage

The ever-growing volumes of heterogeneous data bring with them an assortment of performance, maintenance, and security concerns. To help address these issues, vendors have been steadily incorporating intelligence into their storage solutions. Intelligent storage leverages artificial intelligence (AI) and other advanced technologies to proactively manage systems, optimize performance, and address potential issues before they occur.

An intelligent system continuously learns from its environment and automatically adjusts its behavior accordingly. The system collects telemetry data from participating storage systems, aggregates and analyzes the data, and then uses what it learns to maintain and optimize those systems. When effectively implemented, intelligent storage solutions can deliver greater reliability, security, resource utilization, and application performance.

An intelligent storage system relies on a sophisticated analysis engine that leverages AI technologies such as machine learning and deep learning, along with other advanced technologies, including predictive analytics. The engine identifies patterns and anomalies in the data in order to predict problems, forecast trends, identify performance issues, and address other potential issues. At the same time, the engine is continuously learning from the collected data, leading to more accurate predictions and subsequently more efficient storage systems.

Computational Storage

In a traditional compute/storage architecture, data travels between the storage device and the computer's memory, where it can be processed in response to application requests. Under normal operations, data moves freely between the two, running into few latency issues and bottlenecks. However, modern workloads such as AI or big data analytics can run into performance issues because the I/O ports that sit between storage and memory have limited bandwidth and can't keep up with the demand, resulting in bottlenecks that slow response times.

To address this issue, several vendors now offer computational storage solutions that move at least some of the processing to the storage platform itself, an approach sometimes referred to as in-situ processing. Computational storage brings storage and compute resources closer together within the storage layer, where the data can be preprocessed on behalf of the server. Not only does this shorten the data access path and reduce traffic flow—and the latencies that come with it—but the computational components can also take advantage of the parallel processing capabilities inherent in a storage solution, leading to even faster performance.

Computational storage can potentially benefit any latency-sensitive application that processes large quantities of data. It might also benefit edge computing and Internet of Things (IoT) scenarios, where compute resources are often limited by size. For example, you might aggregate a massive dataset in-situ and then send only the aggregated results to the server's memory for additional processing. In this way, you reduce the amount of data that must pass through the I/O

ports, while minimizing the impact on compute resources, which in turn frees them up for other workloads.
Although several vendors now offer computational storage systems, the industry is still very young. It's not uncommon to run into integration issues because of the differences in implementations. Fortunately, the Storage Networking Industry Association (SNIA) has launched an effort to define interface standards for deploying, provisioning, managing, and securing computational storage devices.

Storage-Class Memory

Another modern technology that's generating a lot of buzz right now is storage-class memory (SCM), a type of memory that's nearly as fast as dynamic random-access memory (DRAM) but like NAND flash is nonvolatile (that is, it can retain data even if unplugged from power). SCM also comes at a lower per-byte cost than DRAM, while substantially outperforming NAND. It might even deliver greater endurance than NAND.

Like computational storage, SCM is still a young technology, but has a great deal of momentum behind it, with Intel at the forefront. You'll likely see SCM also referred to as persistent memory, PMEM, or P-MEM. Some sources distinguish between SCM and persistent memory, depending on how the technology is implemented, but such inconsistencies are common in a nascent industry like SCM, and no doubt the industry will eventually settle on a common nomenclature.

Discussions around SCM often center around the idea of a new tier in the memory/storage hierarchy, with SCM modules sitting between DRAM and NAND flash. Like DRAM, the SCM device is byte-addressable and can connect directly to the server's memory space, providing an effective way to support latency-sensitive applications that need more memory than DRAM can practically provide.

By bridging the gap between traditional memory and storage, SCM makes it possible for applications to access large datasets through the system memory space, resulting in substantially faster read and write operations. At the same time, an SCM device can support block-level access like NAND flash, providing greater versatility than either DRAM or NAND

Initially, much of the focus on the SCM technology has been on devices that can be used as storage cache or to replace flash solid-state drives (SSDs). Intel led the way in this effort with its line of Optane DC SSDs, which work much like NAND flash SSDs, but deliver greater performance.

More recently, Intel introduced its Optane DC persistent memory modules. These plug directly into standard dual in-line memory module (DIMM) slots. An Optane module can store up to 512 GB of data, far exceeding today's DRAM, although such capacities might become more common for DRAM in the near future. In this way, the module can serve as a storage tier between DRAM and NAND flash, moving us closer to the original SCM vision.

It's also possible to use SCM in place of DRAM. Although the SCM modules are slower, their ability to persist data makes them well suited as bootable devices. For example, you might use SCM for a production server that needs to be up and running as quickly as possible after a planned or unplanned restart.

Optane DC persistent memory is based on 3D XPoint technology, which represents a joint effort between Intel and Micron Technology. Micron recently released its first product based on 3D XPoint, the X100 SSD, following the same path as Intel by first introducing an SSD. However, 3D XPoint is not the only SCM effort under way. Other vendors are working on their own solutions based on technologies such as magnetoresistive RAM (MRAM) and nanotube RAM (NRAM).

Moving toward the Future
There is, of course, much more to each of these technologies than what I can cover in a single article, and there are plenty of other emerging technologies as well, such as 5D storage, which uses ultra-fast laser technology to embed data on fused silica glass.

Convergence and Composability
In this article in the storage series, Robert Sheldon explains infrastructure options that simplify administration and improve resource utilization. He discusses the differences and benefits of converged, hyperconverged, and composable infrastructures.

Converged Infrastructure
A converged infrastructure consolidates compute, storage, network, and virtualization technologies into an integrated platform optimized for specific workloads, such as a database management system or virtual desktop infrastructure (VDI). Each component is prequalified, preconfigured, and assembled into a highly engineered system to provide a complete data solution that's easier to implement and maintain than a traditional infrastructure.
The components that make up a converged infrastructure can be confined to a single rack or span multiple racks, depending on the supported workloads. Each component serves as a building block that works in conjunction with the other components to create a unified, integrated platform.

Despite this integration, each component remains a discrete resource. In this way, you can add or remove individual components as necessary, while still retaining the infrastructure's overall functionality (within reason, of course). In addition, you can reuse a removed component for other purposes, an important distinction from HCI.
A converged infrastructure can support a combination of both hard-disk drives (HDDs) and solid-state drives (SSDs), although many solutions have moved toward all-flash storage. The storage is typically attached directly to the servers, with the physical drives pooled to create a virtual storage area network (SAN).

Hyperconverged Infrastructure

The HCI platform takes convergence to the next level, moving from a hardware-based model to a software-defined approach that abstracts the physical compute, storage, and (more recently) network components and presents them as shared resource pools available to the virtualized applications.
Hyperconvergence can reduce data center complexity even further than the converged infrastructure while increasing scalability and facilitating automation. HCI may add features such as data protection, data deduplication, intelligent management, and cloud bursting.
An HCI solution is typically made up of commodity compute, storage, and network components that are assembled into self-contained and highly integrated nodes. Multiple nodes are added together to form a cluster, which serves as the foundation for the HCI platform. Because storage is attached directly to each node, there is no need for a physical SAN or network area storage (NAS).
Each node runs a hypervisor, management software, and sometimes other specialized software, which work together to provide a unified platform that pools resources across the entire infrastructure. To scale out the platform, you simply add one or more nodes.
Initially, HCI targeted small-to-midsized organizations that ran specific workloads, such as VDI. Since then, HCI has expanded its reach to organizations of all sizes, while supporting a broader range of workloads, including database management systems, file and print services, email servers, and specialized solutions such as enterprise resource planning.

Composable Infrastructure

A composable infrastructure pushes beyond the limits of convergence and hyperconvergence by offering a software-defined infrastructure made up entirely of disaggregated, commodity components. The composable infrastructure abstracts compute, storage, and network resources and presents them as a set of unified services that can be allocated on-demand to accommodate fluctuating workloads. In this way, resources can be dynamically composed and recomposed as needed to support specific requirements.

You'll sometimes see the term composable infrastructure used interchangeably with software-defined infrastructure (SDI) or infrastructure as code (IaC), implying that they are one in the same, but this can be misleading. Although a composable solution incorporates the principles of both, it would be more accurate to say that the composable infrastructure is a type of SDI that facilitates development methodologies such as IaC. In this sense, the composable infrastructure is an SDI-based expansion of IaC.

Regardless of the labeling, the important point is that a composable infrastructure provides a fluid pool of resources that can be provisioned on-demand to accommodate multiple types of workloads. A pool can consist of just a couple compute and storage nodes or be made up of multiple racks full of components. Organizations can assemble their infrastructures as needed, without the node-centric restrictions typical of HCI.

Converged to Composable and Beyond

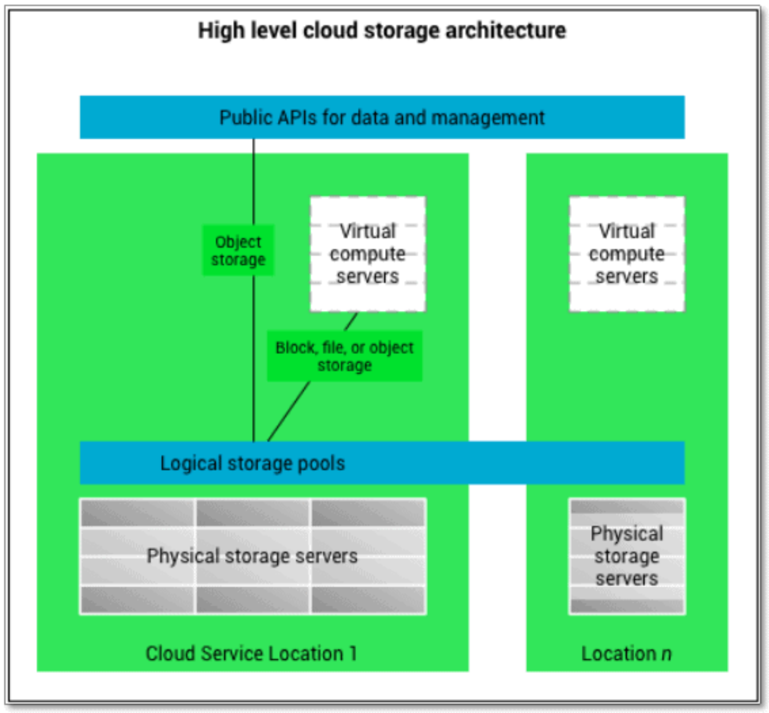
Choosing a converged or composable infrastructure is not an all-or-nothing prospect. You can mix-and-match across your organization as best meets your requirements. For example, you might implement HCI systems in your satellite offices but continue to use a traditional infrastructure in your data center. In this way, you can easily manage the HCI platforms remotely and reduce the time administrators need to spend at those sites, while minimizing the amount of space being used at those locations.

Data and storage will play a pivotal role in any infrastructure decisions. To this end, you must take into account multiple considerations, including application performance, data protection, data quantities, how long data will be stored, expected data growth, and any other factors that can impact storage. You should also consider integration with other systems, which can range from monitoring and development tools to hybrid and public cloud platforms.

Cloud Storage Architecture

Cloud storage is delivered through a virtualized infrastructure that logically pools physical storage resources and presents them as services that are accessible through a centralized portal for easy allocation. Customers can also interact with the storage pool through a public API that facilitates data access and management. The storage pool can span multiple servers or even multiple locations, with the data itself distributed across drives. The combination of distributed data and the operation's inherent redundancy delivers a high degree of fault tolerance and reliability.

Figure 1 provides a conceptual overview of the cloud storage architecture. The exact implementation varies by service provider, with new technologies continuously emerging and existing ones evolving. Even so, the figure should give you a sense of how the physical storage resources are abstracted into logical storage pools that can then be consumed as part of a data management strategy.



The Tempting World of Cloud Storage

Many organizations turn to cloud storage because of its pay-for-use, consumption-based payment structure, which enables them to move from a capital expense (CapEx) model to an operating expense (OpEx) model. Not only does this eliminate the steep up-front costs that come with on-premises storage solutions, but it also avoids having to over-provision storage resources to accommodate fluctuating workloads or anticipated increases in data volumes.

With cloud storage, customers pay only for the storage they need when they need it. There might be base fees attached to the service, but these may be trivial compared to purchasing, housing, and maintaining a system on-premises. That said, cloud services costs can quickly add up, as I'll explain in a bit.

The Dark Side of Cloud Storage

Despite the protections that cloud storage providers have in place, security remains one of the top reasons that organizations are hesitant to put their data out on the cloud, in no small part because of the number of big-name breaches to make the headlines. Even under the best circumstances,

storing data in the cloud increases the attack surface area, with data crossing more networks, distributed to more locations, and replicated more frequently. The larger the attack surface area, the higher the chance that the data can be compromised.
Cloud providers, especially the big players, have big bullseyes painted on them, with international cybercriminals using the most sophisticated weaponry available to try to penetrate infrastructure defenses. In addition, providers must guard against internal threats, which might come in the form of espionage, rogue employees, or careless in-house practices.

Private and Hybrid Cloud Storage

Although many organizations have concerns about cloud storage, they like the service-based delivery model that the cloud affords, which is why some deploy private or hybrid clouds. A private cloud is a dedicated platform that offers storage and compute resources as services, similar to a public cloud. The components that make up a private cloud infrastructure might be housed on-premises or in a colocation facility, and in either case, the organization has complete control over the components.

A private cloud offers some of the same flexibility, scalability, efficiency, and ease-of-use as a public cloud, although not nearly to the same degree, especially when it comes to flexibility and scalability. Even so, the private could be a useful solution for organizations that want more control over their storage or that have strict security and compliance requirements, which is often the case for organizations such as government agencies, financial institutions, or healthcare companies.

Making the Most of Cloud Storage

Initially, cloud storage was seen as a vehicle for reducing CapEx for smaller organizations, allowing them to store their data on a public cloud platform, without the paying for storage they did not need. Now organizations of all sizes and types are leveraging cloud storage, taking advantage of the different deployment options to support a wide range of workloads, such as disaster recovery, file archiving, DevOps development processes, seasonal fluctuations, Internet of Things (IoT) analytics, or any number of other possible use cases.

Data Security and Privacy

Securing Data and Protecting Privacy

For many organizations, their most important asset is their data, the bulk of which must be protected against unauthorized access. The data might include intellectual property, legal documents, passwords, encryption keys, personally identifiable information (PII), or a variety of other sensitive material.

An organization that handles sensitive data should have a comprehensive data protection strategy in place to contend with potential threats. Unfortunately, the exact meaning of data protection is not always clearcut and can vary depending on usage and circumstances. It might refer to securing data, safeguarding privacy, protecting storage systems, implementing disaster recovery (DR), or any combination of these.

According to the SNIA (formerly the Storage Networking Industry Association), data protection is the "assurance that data is not corrupted, is accessible for authorized purposes only, and is in compliance with applicable requirements." In other words, data protection goes beyond just encrypting data or guaranteeing its availability. Data protection ensures that the data remains viable, is safeguarded against all unauthorized access at all times, and is controlled in a way that adheres to applicable compliance laws and regulations, e.g., local, provincial, and federal.

The Cybersecurity Threat Landscape

Data is not only an asset. It's a burden. A data breach can lead to lost revenue, stiff penalties, downtime, legal liabilities, loss of intellectual property, unexpected expenses, and a tarnished reputation from which a company might never recover. No organization is immune to the potential threats that await them, from both inside and outside their domains.

External threats can come from governments, organized crime, terrorists, cybercriminals, competitors, or everyday hackers looking for a bit of sport or profit. And threats can arrive in many forms, often implemented through social engineering strategies that attempt to introduce malware or ransomware or steal user credentials.

Implementing a Data Protection Strategy

To ensure data security and privacy, you need a comprehensive plan that specifies how data will be protected both at rest and in motion. As part of this process, you should develop policies that define where data can be stored, who can access it, and what levels of protection the data requires. The policies should also address such issues as when data is deleted, what happens when an employee is terminated, how to handle a data breach and any other issues related to data protection.

Another important part of the planning process is to conduct a thorough assessment of your current data environment to identify potential risks and the steps that must be taken to mitigate those risks. You need to know where sensitive data is located, how it's being used, and who can access it. You should also look for issues such as whether sensitive data is being transmitted as cleartext, credentials are being sent in an unencrypted format, or users are accessing internal web services via insecure HTTP.

Protecting Data and Privacy

Data protection must take into account both physical and operational security. Physical security ensures that unauthorized individuals cannot access the physical structures where the data is housed or the equipment within those structures. It also protects against circumstances that could lead to data loss, such as power failures or natural disasters. To implement physical security, an organization might employ backup and restore protocols, CCTV monitoring, biometric readers, geofencing, backup generators, and numerous other protections.

Organizations must also protect the individual systems within their secure structures, such as servers or workstations. No one on the inside should be able to walk off with equipment or get at their internal workings unless they're authorized to do so. IT teams must also take steps to protect portable devices that leave the premises, such as laptops, tablets, or cell phones. This typically means implementing a mobile device management strategy that supports such features as remote lock or remote wipe.

The Ongoing Challenges of Data Protection

To implement effective data protections, an organization must take into account the entire data lifecycle, regardless of how the data is being used or where it resides—whether on a remote workstation, on a mobile device, in a data center, on a cloud platform, at a remote facility, or on a server in an office corner. Data protection must be a unified effort that moves beyond infrastructure boundaries to ensure that data is secure, and privacy is protected at all times and under all circumstances.

UNIT IV Completed