Introduction to Storage Technologies

CITRIX®

Contents

Overview

This document is an introduction to Disk Storage technologies and its terminology. Within this document basic disk and storage architectures as well as storage protocols and common fault tolerance technologies will be discussed. It is not intended as a comprehensive guide for planning and configuring storage infrastructures, nor as a storage training handbook.

Due to scope, this guide provides some device-specific information. For additional device- specific configuration, Citrix suggests reviewing the storage vendor's documentation, the storage vendor's hardware compatibility list, and contacting the vendor's technical support if necessary.

For design best practices and planning guidance, Citrix recommends reviewing the Storage Best Practices and Planning Guide (http://support.citrix.com/article/CTX130632)

Disks

While all kinds of disk drives offer the ability to store large amounts of data persistently, there are major differences in the way the data is actually stored or accessed in between the two existing technologies. Within this chapter we will discuss both technologies and their pros and cons respectively.

Access Pattern

Some of the most important and discussed terms within the storage technology area are the access pattern of applications or VM workloads using storage sub-systems. Besides the distinction of reads and writes and their respective ratio, a major differentiator is sequential access vs. random access. The following diagram outlines both access scenarios:

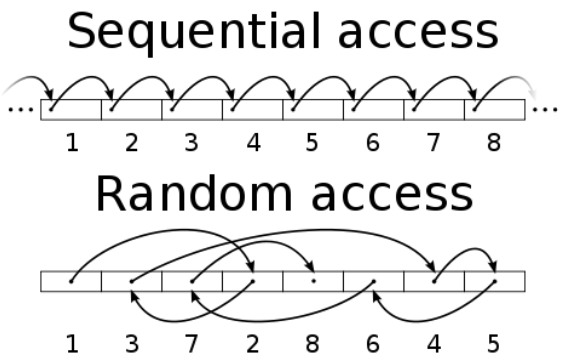


Figure 1: Sequential vs. Random access[1]

Hard Disk Drives (HDDs)

Hard disk drives or HDDs are the traditional variation of disk drives. These kinds of disks consist of "rotating rigid platters on a motor-driven spindle within a protective enclosure. Data is magnetically read from and written to the platter by read/write heads that float on a film of air above the platters"[2].

[1] http://en.wikipedia.org/wiki/File:Random_vs_sequential_access.svg
[2] http://en.wikipedia.org/wiki/Hard_drives

A typical read/write operation consists of the following steps:

Step	Description
1	The Disk controller translates a logical address into a physical address (cylinder, track, and sector). The request is a matter of a few tens of nanoseconds, the command decoding and translating can take up to 1 ms.
2	The head is moved by the actuator to the correct track. This is called seek time, the average seek time is somewhere between 3.5 and 10 ms
3	The rotational motor makes sure that the correct sector is located under the head. This is called rotational latency and it takes from 5.6 ms (5400 rpm) to 2 ms (15000 rpm). Rotational latency is thus determined by how fast the rotational motor spins.
4	The data is then read or written. The time it takes is dependent on how many sectors the disk has to write or read. The rate at which data is accessed is called the media transfer rate (MTR).
5	If data is read, the data goes into disk buffer, and is transferred by the disk interface to the system.

Table 1: Typical HDD read/write operation[3]

Based on the process outlined above the performance of a disk seen from an application or VM level point of view, depends on the amount and the pattern of the data read or written. In case many small files need to be accessed that are distributed across all parts of the disk, the disk will spend a large amount of time seeking for the blocks rather than actually reading or writing data (random access). Opposed to this the disk will just seek once and then start reading / writing in case of a single file stored at contiguous disk blocks (sequential access). Therefore the performance of a disk is measured in MB/second and I/O Operations Per Second (IOPS).

While all spinning hard disks share the same electromagnetic approach, different implementations of this technology exist on the market, which differ in terms of performance, cost and reliability. In this regards the most important differentiation is in between Serial Attached SCSI (SAS) disks and Serial Advanced Technology Attachment (SATA) disk.

SATA	SAS
Build for desktop use	Build for Enterprise use
Low cost	High cost
~75 IOPS	~200 IOPS
Latency 13ms @7.200 RPM	Latency 6ms @ 15.000 RPM
Up to 300.000h MTBF	Over 900.000h MTBF

Table 2: SATA vs. SAS

Note: Disk interface buses are explained in more detail on page 9.

[3] http://www.anandtech.com/show/2105/1

Solid State Disks

In contrast to traditional hard disks, Solid State Disks (SSDs) "use microchips which retain data in non-volatile memory chips (flash) and contain no moving parts. Compared to electromechanical HDDs, SSDs are typically less susceptible to physical shock, are silent, have lower access time and latency (typically <1ms), and have higher I/O rates (typically >3.000), but are more expensive per gigabyte (GB) and typically support a limited number of writes over the life of the device. SSDs use the same interface as hard disk drives, thus easily replacing them in most applications."[4]

SSDs can be either based on multi-level cells (MLC), which are lower priced, slower and less reliable than single-level cells (SLC), which are primarily used for high-end SSDs. For scenarios with ultra high performance requirements typical flash disks can be combined with DRAM arrays, which hold all active data during normal operations. In case of a power outage, all data is written to the "traditional" flash modules using a backup battery.

In contrast to traditional Hard Disk Drives (HDDs) SSD have a limited number of write cycles a single flash cell or a complete SSD drive is able to sustain (write endurance). Typical MLC cells can perform approx. 3.000 – 5.000, typical SLC cells approx. 100.000 and special SLC cells up to 5 million write cycles. Beyond that breaking point, flash cells cannot take any additional write and become read-only. To increase the life time of a SSD, wear leveling was developed, which automatically distributes writes across all cells of a SSD evenly. Wear leveling will become active as soon as all blocks of the SSD have been written once.

Comparing SSDs and ordinary (spinning) HDDs is difficult. "Traditional HDD benchmarks are focused on finding the performance aspects where they are weak, such as rotational latency time and seek time. As SSDs do not spin, or seek, they may show huge superiority in such tests. However, SSDs have challenges with mixed reads and writes, and their performance may degrade over time. SSD testing must start from the (in use) full disk, as the new and empty (fresh out of the box) disk may have much better write performance than it would show after only weeks of use."[5]

	HDD	SSD New (Empty)	SSD Working (Full)
Random Read IO	Challenging	Very Fast	Very Fast
Random Write IO	Challenging	Very Fast	Impacted
Sequential Read	Fast	Very Fast	Very Fast
Sequential Write	Fast	Very Fast	Impacted
Mixed Reads and Writes	Fast	Challenging	Impacted

Figure 2: Performance aspects of storage devices.[6]

[4] http://en.wikipedia.org/wiki/Ssd

[5,6] http://www.stec-inc.com/downloads/whitepapers/Benchmarking_Enterprise_SSDs.pdf

The reason for full SSDs being considerably slower than empty SSDs is related to the basic functionality of a SSD. "Due to the nature of Flash memory's operation, data cannot be directly overwritten as it can in a hard disk drive. When data is first written to an SSD, the cells all start in an erased state so data can be written directly using pages at a time (often 4-8 kilobytes (KB) in size). The SSD controller on the SSD, which manages the Flash memory and interfaces with the host system, uses a logical to physical mapping system known as logical block addressing (LBA) and that is part of the Flash translation layer (FTL). When new data comes in replacing older data already written, the SSD controller will write the new data in a new location and update the logical mapping to point to the new physical location. The old location is no longer holding valid data, but it will eventually need to be erased before it can be written again.

Once every block of an SSD has been written one time, the SSD controller will need to return to some of the initial blocks which no longer have current data (also called stale blocks).

"Data is written to the Flash memory in units called pages (made up of multiple cells). However the memory can only be erased in larger units called blocks (made up of multiple pages). If the data in some of the pages of the block are no longer needed or needs to be overwritten, all of the other pages with good data in that block must be read and re-written. "[7] This intensification of writes caused by re-writing data which has not been modified is called write amplification and may occupy large portions of the available performance capacity of a SSD drive. Write amplification is typically caused by the SSD internal Garbage Collection process, but may happen during normal disk operations as well. Common amplification ratios are typically in between 1 (best) and 20 (worst case).

In order to minimize the write amplification and the related performance impact different technologies have been developed. Most commonly known is the TRIM command, which marks SSD data pages that are no longer considered in use and can be wiped SSD internally. This allows "the SSD to handle garbage collection overhead, which would otherwise significantly slow down future write operations to the involved blocks, in advance"[8]. In order to enable TRIM, all components involved in storage operations such as the operating system, the RAID controller and the SSD itself need to have build-in TRIM support. In addition most SSD drives include a garbage collection process which increases the results of TRIM, by erasing stale blocks.

[7] http://en.wikipedia.org/wiki/Garbage_collection_(SSD)
[8] http://en.wikipedia.org/wiki/TRIM

Disk Interface Buses

The following section outlines the most common disk interfaces.

- **Advanced Technology Attachment (ATA):** This is a legacy interface more common in home computing, and can support different kinds of devices, such as hard drives and DVD burners. There is a restriction of 2 devices per cable. This is also known as a parallel ATA, since there is a wire for each bit of information in the interface cable, making it very wide. The disk drives that are attached to an ATA host adapter are usually called IDE drives (Integrated Drive Electronics).

- **Serial ATA (SATA):** Is an enhancement to ATA, which allows for changing drives without shutting down (hot swap), faster transfer speeds, and thinner cabling.

Traditional hard disks that attach either through ATA or SATA, have their disk platters spinning at usually 5.400 or 7,200 revolutions per minute (RPM). Remember that the disk spin speed is one important measure of the disk's access time. Within enterprise environments the following disk interface types are more common than the afore mentioned:

- **Small Computer System Interface (SCSI):** An interface standard that is not compatible with ATA or IDE drives. Modern versions of SCSI affords up to 16 devices per cable including the host adapter. Although the layout looks like ATA, none of the components are interchangeable.

- **Serially Attached SCSI (SAS):** A point-to-point serial protocol that replaces the parallel SCSI bus technology mentioned above. It uses the standard SCSI command set, but is currently not faster than parallel SCSI. In the future, speeds are expected to double, and there will also be the ability to use certain (slower) SATA drives on a SAS bus.

Traditional hard disks that attach through SCSI or SAS usually spin at 10,000 or 15,000 RPM. Because of this, and the more complicated electronics, SCSI components are much more expensive than S/ATA. But SCSI disks are renowned for their speed of access, and data transfer.

The following table comparison between the common disk interface bus technologies:

Name	Raw bandwidth (Mbit/s)	Transfer speed (MByte/s)
PATA	1.064	133
SATA (rev 1)	1.500	150
SATA (rev 2)	3.000	300
SATA (rev 3)	6.000	600
SAS 150	1.500	150
SAS 300	3.000	300
SAS 600	6.000	600

Table 3: Disk Interface comparison[9]

[9] http://en.wikipedia.org/wiki/Serial_ATA#Comparison_with_other_buses

Citrix Recommendations

The following table summarizes this chapter and provides related recommendations.

(++ = very positive / O = balanced / -- very negative)

	Hard Disk Drives		Solid State Drives
	SATA	**SAS**	
Price	++	+	-
Data access times	-	+	++
Energy consumption	-	-	+
Reliability	O	++	O
Random Read	-	O	++
Random Write	-	O	++ (initially) O (when full)
Sequential Read	O	+	++
Sequential Write	O	+	++ (initially) O (when full)
Mixed Read Write	-	+	++ (initially) O (when full)
Best suited for	Data w/o high performance requirements such as: - PVS vDisk store	Data w/ high performance requirements such as: - PVS Write Cache - XenServer storage (IntelliCache) Note: When combining multiple SAS drives into special RAID arrays such as level 10, performance level equal or higher than SSD can be achieved	
Not recommended for	Data w/ high performance requirements such as: - PVS Write Cache - XenServer storage (IntelliCache)		

Storage Architectures

Within the world of storage, three main architectures have evolved:

- Directly Attached Storage (DAS)
- Storage Area Network (SAN)
- Network Attached Storage (NAS)

These three architectures are outlined within the following diagram:

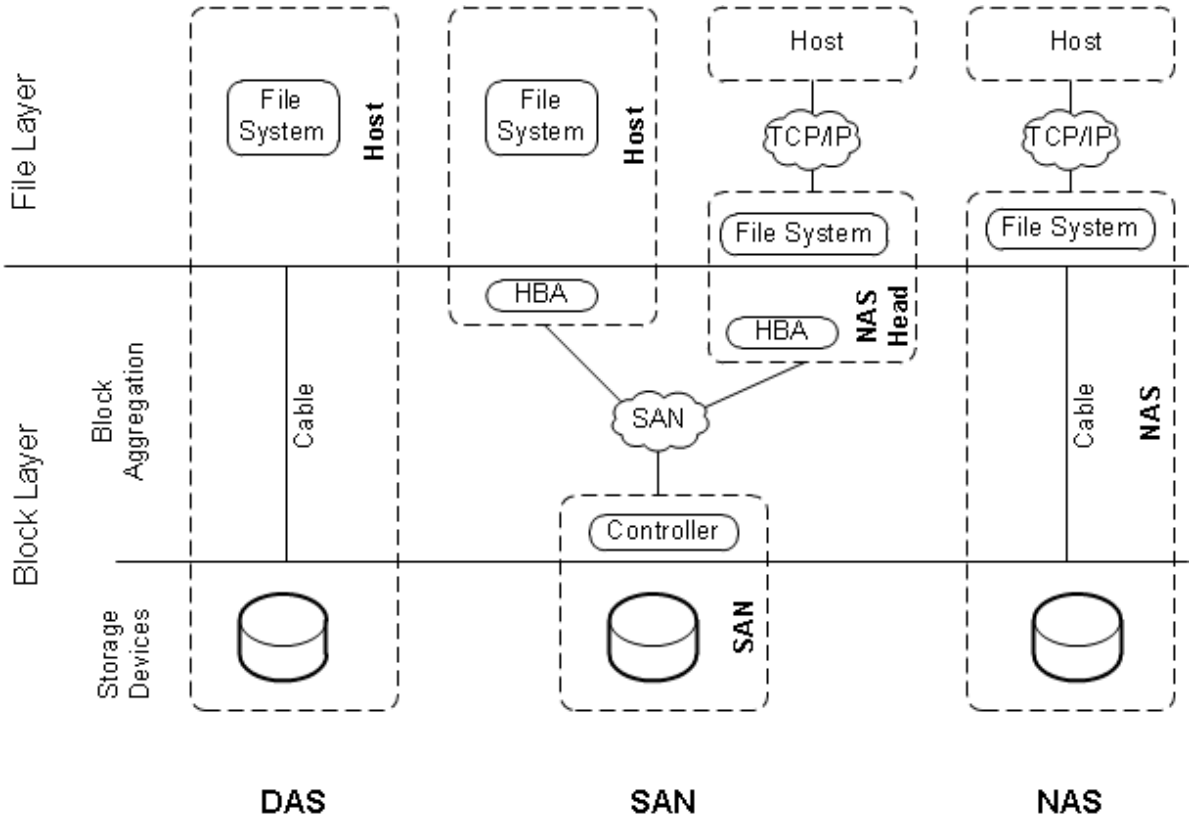


Figure 3: Storage architectures

In addition to these three storage architectures, there are also other models that utilizes portions of the aforementioned core models.

Directly Attached Storage (DAS)

A Directly Attached Storage is a storage sub-system that is directly attached to a server or workstation using a cable. It can be a hard disk directly built into a computer system or a disk shelf with multiple disks attached by means of external cabling. Contrary to local hard disks, disk shelves require separate management. In some cases, storage shelves can be connected to multiple servers so the data or the disks can be shared (i.e. for fault tolerance). Additionally DAS disk shelves are able to 'hot swap' failed disks and to rebuild disk from parity on other disks in most cases. Common storage protocols for DAS are SATA, SAS and Fibre Channel.

Network Attached Storage (NAS)

"Network-attached storage (NAS) is file-level computer data storage connected to a computer network providing data access to heterogeneous clients. NAS not only operates as a file server, but is specialized for this task either by its hardware, software, or configuration of those elements. NAS systems are networked appliances which contain one or more hard drives, often arranged into logical, redundant storage containers or RAID arrays. Network-attached storage removes the responsibility of file serving from other servers on the network. They typically provide access to files using standard Ethernet and network file sharing protocols such as NFS, SMB/CIFS, or AFP".[10]

Storage Area Network (SAN)

"A storage area network (SAN) is a dedicated storage network that provides access to consolidated, block level storage. SANs primarily are used to make storage devices (such as disk arrays, tape libraries, and optical jukeboxes) accessible to servers so that the devices appear as locally attached to the operating system. The cost and complexity of SANs dropped in the early 2000s, allowing wider adoption across both enterprise and small to medium sized business environments. A SAN alone does not provide the "file" abstraction, only block-level operations".[11] A SAN typically has its own dedicated network of storage devices that are generally not accessible through the regular network by regular devices. In order to connect a device to the SAN network a specialized extension card called Host Bus Adapter (HBA) is required.

Hybrid / NAS Heads

A NAS head refers to a NAS which does not have any on-board storage, but instead connects to a SAN. In effect, it acts as a translator between the file-level NAS protocols (NFS, CIFS, etc.) and the block-level SAN protocols (Fibre Channel Protocol, iSCSI). Thus it can combine the advantages of both technologies and allows computers without a Host Bus Adapter (HBA) to connect to the centralized storage.

[10] http://en.wikipedia.org/wiki/Network-attached_storage
[11] http://en.wikipedia.org/wiki/Storage_area_network

Tiered storage

Tiered storage is a data storage environment consisting of two or more kinds of storage delineated by differences in at least one of these four attributes: Price, Performance, Capacity and Function.

In mature implementations, the storage architecture is split into different tiers. Each tier differs in the:

- Type of hardware used
- Performance of the hardware
- Scale factor of that tier (amount of storage available)
- Availability of the tier and policies at that tier

A very common model is to have a primary tier with expensive, high performance and limited storage. Secondary tiers typically comprise of less expensive storage media and disks and can either host data migrated (or staged) by Lifecycle Management software from the primary tier or can host data directly saved on the secondary tier by the application servers and workstations if those storage clients did not warrant primary tier access. Both tiers are typically serviced by a backup tier where data is copied into long term and offsite storage.

Within this context, two terms should be mentioned:

- ILM – Information Lifecycle Management refers to a wide-ranging set of strategies for administering storage systems on computing devices.
  http://en.wikipedia.org/wiki/Information_Lifecycle_Management
- HSM – Hierarchical Storage Management is a data storage technique which automatically moves data between high-cost and low-cost storage media. HSM systems exist because high-speed storage devices, such as hard disk drive arrays, are more expensive (per byte stored) than slower devices, such as optical discs and magnetic tape drives. HSM can be implemented as out-of-band process (runs at fixed intervals) or as dynamic tiering (constantly running process).
  http://en.wikipedia.org/wiki/Hierarchical_storage_management

Storage Admission Tier (SAT)

The goal of Storage virtualization is to turn multiple disk arrays, made by different vendors, scattered over the network, into a single monolithic storage device, which can be managed uniformly.

The Storage Admission Tier (SAT) is a tier put in front of the primary tier, as the way into the storage. This affords a way to manage access, and policies in a way that can virtualize the storage.

SAT should conform to the 'Virtualize, Optimize & Manage' paradigm (VOM):

- **Virtualize:** At the SAN layer, the amalgamation of multiple storage devices as one single storage unit greatly simplifies management of storage hardware resource allocation. At the NAS layer, the same degree of virtualization is needed to make multiple heterogeneous file server shares appear as at a more logical level, abstracting the NAS implementations from the application tier.
- **Optimize:** Can include things like compression, data de-duplication (http://en.wikipedia.org/wiki/Data_deduplication) and organizational decisions of data placement (which tier should the data be placed?)
- **Management:** To control policies, security and access control (including rights management) from the entry and exit point of the data to and from the storage network.

File Access Networks (FAN)

The combination of the Storage Access Tier (SAT), the Tiered Storage Model and NAS/SAN are known as the File Area Network (FAN). As of this writing, the concept of FAN cannot be seen in any mainstream products, but the concept is introduced for completeness.

Storage Transport Protocols

Within this chapter the most commonly used storage transport protocols will be discussed in detail.

CIFS / SMB

"The Server Message Block (SMB) protocol, also known as Common Internet File System (CIFS) operates as an application-layer (OSI layer 7) network protocol mainly used to provide shared access to files, printers, serial ports, and miscellaneous communications between nodes on a network. It also provides an authenticated inter-process communication mechanism".[12]

The Server Message Block protocol can run atop the Session (and underlying) network layers in several ways:

- directly over TCP port 445
- via the NetBIOS API, which in turn can run on several transports:
    - on TCP and UDP ports 137, 138, 139

This protocol is most commonly used within Windows based environments.

"SMB works through a client-server approach, where a client makes specific requests and the server responds accordingly. One section of the SMB protocol specifically deals with access to file systems, such that clients may make requests to a file server; but some other sections of the SMB protocol specialize in inter-process communication (IPC). The Inter-Process Communication (IPC) share or IPC$ is a network share on computers running Microsoft Windows. This virtual share is used to facilitate communication between processes and computers over SMB, often to exchange data between computers that have been authenticated. Almost all implementations of SMB servers use NT Domain / Active Directory authentication to validate user-access to resources.".[8]

After the initial design of the SMB protocol by IBM, it has been steadily developed by Microsoft, which released new versions as part of its Windows release cycle.

Windows Version	SMB Version
Prior Windows 2008 / Vista	SMB 1.0
Windows 2008 / Vista	SMB 2.0
Windows 2008 R2 / 7	SMB 2.1

SMB2 reduced the 'chattiness' of the SMB 1.0 protocol by reducing the number of commands and subcommands from over a hundred to just nineteen. It has mechanisms for pipelining, that is, sending additional requests before the response to a previous request arrives. Furthermore it added the ability to compound multiple actions into a single request, which significantly reduces

[12] http://en.wikipedia.org/wiki/Server_Message_Block

the number of round-trips the client needs to make to the server, improving performance especially over high latency links as a result.[8]

In order to allow two computer systems to communicate using SMB 2.0 or 2.1 it is necessary that both systems are running Windows 2008 / Vista (for SMB 2.0) or Windows 2008 R2 / 7 (for SMB 2.1). In case of mixed scenarios the capabilities of the operating system with the lowest version determine the SMB level.

Opportunistic locking

"Opportunistic locks, or Oplocks, are mechanisms designed to allow clients to dynamically alter their buffering strategy for a given file or stream in a consistent manner to increase performance and reduce network use. The network performance for remote file operations may be increased if a client can locally buffer file data, which reduces or eliminates the need to send and receive network packets. For example, a client may not have to write information into a file on a remote server if the client knows that no other process is accessing the data. Likewise, the client may buffer read-ahead data from the remote file if the client knows that no other process is writing data to the remote file. Oplocks can also be used by applications to transparently access files in the background.

File systems like NTFS support multiple data streams per file. Oplocks are "stream" handle centric, this means the operations apply to the given open stream of a file and in general operations on one stream do not affect Oplocks on a different stream. There are exceptions to this, which will be explicitly pointed out. For file systems that don't support alternate data streams think of "file" when this document refers to "stream". There are four different types of Oplocks".[13]:

- A Level 2 (or shared) Oplock indicates that there are multiple readers of a stream and no writers. This supports read caching.
- A Level 1 (or exclusive) Oplock allows a client to open a stream for exclusive access and allows the client to perform arbitrary buffering. This supports read and write caching
- A Batch Oplock (also exclusive) allows a client to keep a stream open on the server even though the local accessor on the client machine has closed the stream. This supports read, write and handle caching.
- A Filter Oplock (also exclusive) allows applications and file system filters which open and read stream data a way to "back out" when other applications/clients try to access the same stream. Filter Oplock support was added in Windows 2000. This supports read and write caching.

[13] http://msdn.microsoft.com/en-us/library/cc308442.aspx

NFS

The Network File System (NFS) protocol is, similar to CIFS / SMB, an application-layer (OSI layer 7) network protocol. In general NFS is a low complexity protocol (compared to SMB), which only allows access to files over an Ethernet network. Within RFC1813 NFS is described as follows: "NFS servers are dumb and NFS clients are smart. It is the clients that do the work required to convert the generalized file access that servers provide into a file access method that is useful to applications and users. The NFS protocol assumes a stateless server implementation. Statelessness means that the server does not need to maintain state about any of its clients in order to function correctly. Stateless servers have a distinct advantage over stateful servers in the event of a crash. With stateless servers, a client need only retry a request until the server responds; the client does not even need to know that the server has crashed."[14]

NFS was initially developed by Sun. Latest versions of NFS have been developed by the Internet Engineering Task Force (IETF).

NFS Version	Release
1	Sun internal
2	March 1989
3	June 1995
4	April 2003
4.1	January 2010

NFS is most frequently used within Linux or Unix environments, but is also available for other platforms such as Windows (Windows Services for Unix / NFS Client or Server) or Apple Mac OS.

Initially NFS was based on UDP for performance reasons. Starting with version 3 NFS added support for TCP/IP based networks. While CIFS / SMB is user-centric, which means that authentication and authorization happens at the user level, NFS (until version 4 which includes Kerberos authentication support) is computer centric. In many cases NFS version 3 is still the latest version supported, which means Kerberos authentication is not supported and as such traffic is not encrypted. Storage traffic is transmitted as clear text across the LAN. Therefore, it is considered best practice to use NFS storage on trusted networks only and to isolate the traffic on separate physical switches or leverage a private VLAN.

The latest NFS standard (v4.1) added support for parallel NFS (pNFS) which allows clients to access storage devices directly and in parallel. The pNFS architecture eliminates the scalability and performance issues associated with NFS servers in deployment today. This is achieved by the separation of data and metadata, and moving the metadata server out of the data path as shown in the diagram below.

[14] http://tools.ietf.org/html/rfc1813

Fibre Channel

"Fibre Channel, or FC, is a transport protocol (similar to TCP used in IP networks) which predominantly transports SCSI commands over Fibre Channel networks. Despite its name, Fibre Channel signaling can run on both twisted pair copper wire and fiber-optic cables".[15]

In order to enable a system to access a Fibre Channel network, it is necessary to implement a host bus adapter (HBA). "Fibre Channel HBAs are available for all major open systems, computer architectures, and buses. Some are OS dependent. Each HBA has a unique World Wide Name (WWN), which is similar to an Ethernet MAC address in that it uses an Organizationally Unique Identifier (OUI) assigned by the IEEE. However, WWNs are longer (8 bytes). There are two types of WWNs on a HBA; a node WWN (WWNN), which can be shared by some or all ports of a device, and a port WWN (WWPN), which is necessarily unique to each port".[16]

Fibre Channel can be used in three different topologies:

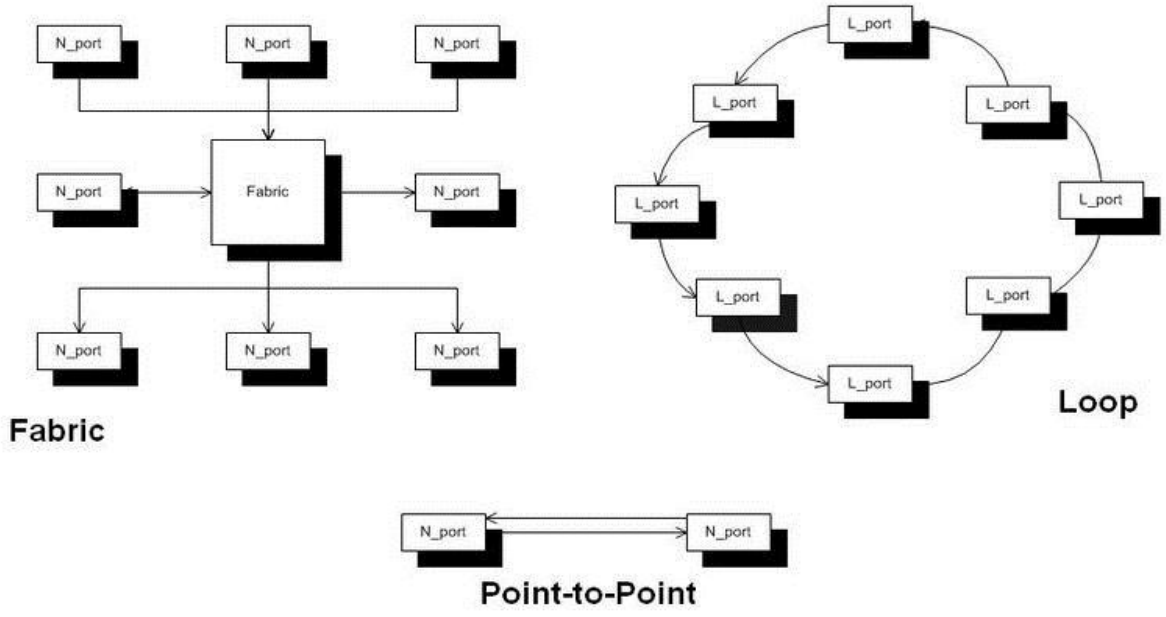


Figure 4: SAN topologies

- **Point-to-Point** (FC-P2P). Two devices are connected back to back. This is the simplest topology, with limited connectivity.
- **Arbitrated loop** (FC-AL). In this design, all devices are in a loop or ring, similar to token ring networking. Adding or removing a device from the loop causes all activity on the loop to be interrupted. The failure of one device causes a break in the ring. Fibre Channel hubs

[15] http://en.wikipedia.org/wiki/Fibre_Channel_Protocol
[16] http://en.wikipedia.org/wiki/Fibre_Channel#Fibre_Channel_Host_Bus_Adapters

exist to connect multiple devices together and may bypass failed ports. A loop may also be made by cabling each port to the next in a ring. A minimal loop containing only two ports, while appearing to be similar to FC-P2P, differs considerably in terms of the protocol.

- **Switched fabric** (FC-SW). All devices or loops of devices are connected to Fibre Channel switches, similar conceptually to modern Ethernet implementations. The switches manage the state of the fabric, providing optimized interconnections.

FC-SW is the most flexible topology, enabling all servers and storage devices to communicate with each other. It also provides for failover architecture in the event a server or disk array fails. FC-SW involves one, or more intelligent switches each providing multiple ports for nodes. Unlike FC-AL, FC-SW bandwidth is fully scalable, i.e. there can be any number of 8Gbps (Gigabits per second) transfers operating simultaneously through the switch. In fact, if using full-duplex, each connection between a node and a switch port can use 16Gbps bandwidth.

Because switches can be cascaded and interwoven, the resultant connection cloud has been called the *fabric*.

Fibre Channel Zoning

"In storage networking, Fibre Channel zoning is the partitioning of a Fibre Channel fabric into smaller subsets to restrict interference, add security, and to simplify management. While a SAN makes available several virtual disks (LUNs), each system connected to the SAN should only be allowed access to a controlled subset of the LUNs. Zoning applies only to the switched fabric topology (FC-SW), it does not exist in simpler Fibre Channel topologies.

Zoning is sometimes confused with LUN masking, because it serves the same goals. LUN masking, however, works on Fibre Channel level 4 (i.e. on SCSI level), while zoning works on level 2. This allows zoning to be implemented on switches, whereas LUN masking is performed on endpoint devices - host adapters or disk array controllers.

Zoning is also different from VSANs, in that each port can be a member of multiple zones, but only one VSAN. VSAN (similarly to VLAN) is in fact a separate network (separate sub-fabric), with its own fabric services (including its own separate zoning)".[17]

[17] http://en.wikipedia.org/wiki/Fibre_Channel_zoning

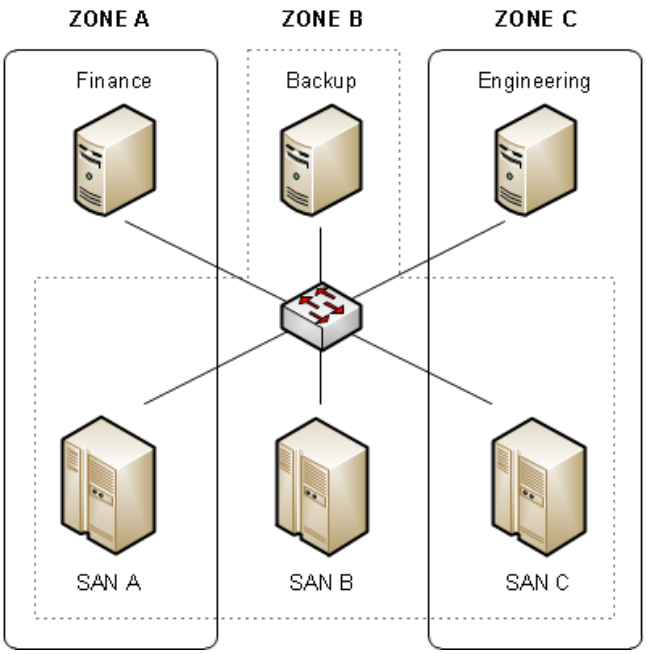


Figure 5: Example of overlapping zones

Zoning can be implemented in one of two ways:

- **Hardware:** Hardware zoning is based on the physical fabric port number. The members of a zone are physical ports on the fabric switch. It can be implemented in the configurations of One-to-one, One-to-many and Many-to-many.

- **Software:** Software zoning is implemented by the fabric operating systems within the fabric switches. They are almost always implemented by a combination of the name server and the Fibre Channel Protocol. When a port contacts the name server, the name server will only reply with information about ports in the same zone as the requesting port. A soft zone, or software zone, is not enforced by hardware (i.e. hardware zoning). Usually, the zoning software also allows you to create symbolic names for the zone members and for the zones themselves. Dealing with the symbolic name or aliases for a device is often easier than trying to use the WWN address.

Fibre Channel over Ethernet

"Fibre Channel over Ethernet (FCoE) is an encapsulation of Fibre Channel frames over Ethernet networks. This allows Fibre Channel to use 10 Gigabit Ethernet networks (or higher speeds) while preserving the Fibre Channel protocol. With FCoE, Fibre Channel becomes another network protocol running on Ethernet, alongside traditional Internet Protocol (IP) traffic. FCoE operates directly above Ethernet in the network protocol stack, in contrast to iSCSI which runs on top of TCP and IP. As a consequence, FCoE is not routable at the IP layer, and will not work across routed IP networks.

Alongside with FCoE a technology called Converged Networks has been introduced. This allows computers to be connected to FCoE with Converged Network Adapters (CNAs), which contain both Fibre Channel Host Bus Adapter (HBA) and Ethernet Network Interface Card (NIC) functionality on the same adapter card. CNAs have one or more physical Ethernet ports. FCoE encapsulation can be done in software with a conventional Ethernet network interface card, however FCoE CNAs offload (from the CPU) the low level frame processing and SCSI protocol functions traditionally performed by Fibre Channel host bus adapters."[18]

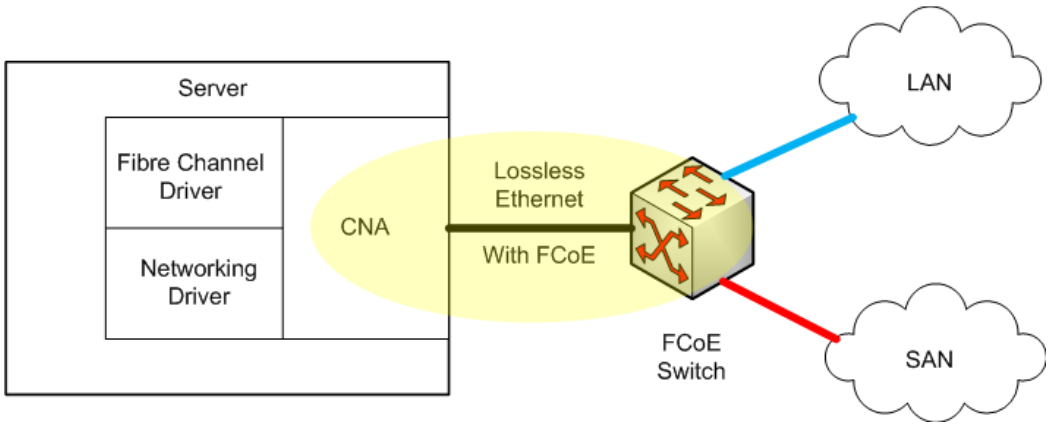


Figure 6: Converged Network

[18] http://en.wikipedia.org/wiki/Fibre_Channel_over_Ethernet

iSCSI

The iSCSI protocol (abbreviation of Internet Small Computer System Interface) which is defined in RFC3720 is a mapping of the regular SCSI protocol over TCP/IP, more commonly over Gigabit Ethernet. Unlike Fibre Channel, which requires special-purpose cabling, iSCSI can be run over long distances using an existing network infrastructure. TCP/IP uses a client/server model, but iSCSI uses the terms initiator (for the data consumer) and target (for the LUN). An initiator falls into two broad types:

- "A Software initiator uses code to implement iSCSI. Typically, this happens in a kernel-resident device driver that uses the existing network card (NIC) and network stack to emulate SCSI devices for a computer by speaking the iSCSI protocol. Software initiators are available for most mainstream operating systems, and this type is the most common mode of deploying iSCSI on computers".[19]

- A hardware initiator uses dedicated hardware that mitigates the overhead of iSCSI, TCP processing and Ethernet interrupts, and therefore may improve the performance of servers that use iSCSI. An iSCSI host bus adapter (HBA) implements a hardware initiator and is typically packaged as a combination of a Gigabit Ethernet NIC, some kind of TCP/IP offload technology (TOE) and a SCSI bus adapter (controller), which is how it appears to the operating system.

iSCSI Naming & Addressing

Each initiator or target is known by an iSCSI Name which is independent of the location of the initiator and target. iSCSI Names are used to identify targets and initiators unequivocally. Furthermore it is used to identify multiple paths in between a target and an initiator.

iSCSI naming can be based on three different formats, whereof the iSCSI Qualified Name (IQN) is most commonly used:

- **iSCSI Qualified Name (IQN)**, format: iqn.yyyy-mm.{reversed domain name}
    - iqn.2001-04.com.acme:storage.tape.sys1.xyz
- **Extended Unique Identifier (EUI)**, format: eui.{EUI-64 bit address}
    - eui.02004567A425678D
- **T11 Network Address Authority (NAA)**, format: naa.{NAA 64 or 128 bit identifier}
    - naa.52004567BA64678D

[19] http://en.wikipedia.org/wiki/Iscsi

The default name "iSCSI" is reserved and is not used as an individual initiator or target name. iSCSI Names do not require special handling within the iSCSI layer; they are opaque and case-sensitive for purposes of comparison.

iSCSI nodes (i.e. the machine that contains the LUN targets) also have addresses. An iSCSI address specifies a single path to an iSCSI node and has the following format:

<domain-name>[:<port>]

Where <domain-name> can be either an IP address, in dotted decimal notation or a Fully Qualified Domain Name (FQDN or host name). If the <port> is not specified, the default port 3260 will be assumed.

iSCSI Security

To ensure that only valid initiators connect to storage arrays, administrators most commonly run iSCSI only over logically-isolated backchannel networks.

For authentication, iSCSI initiators and targets prove their identity to each other using the CHAP protocol, which includes a mechanism to prevent cleartext passwords from appearing on the wire. Additionally, as with all IP-based protocols, IPsec can operate at the network layer. Though the iSCSI negotiation protocol is designed to accommodate other authentication schemes, interoperability issues limit their deployment. An initiator authenticates not to the storage array, but to the specific storage asset (target) it intends to use.

For authorization, iSCSI deployments require strategies to prevent unrelated initiators from accessing storage resources. Typically, iSCSI storage arrays explicitly map initiators to specific target LUNs.

Citrix recommendations

The following table summarizes this chapter and provides related recommendations.

Functional

Protocol	CIFS	NFS	iSCSI	FC
PVS – central vDisk management	X	X	(X)[1]	(X)[1]
PVS – caching of vDisks	(X)[2]	X	X	X
XS – IntelliCache		X		

1: Additional software (clustered file system) required

2: Opportunistic Locking configuration of Windows needs to be changed (see http://blogs.citrix.com/2010/11/05/provisioning-services-and-cifs-stores-tuning-for-performance)

Following the recommendations outlined earlier within this section will be discussed in more detail:

Performance

(++ = very positive / O = balanced / -- very negative)

Protocol	CIFS	NFS	iSCSI	FC
PVS – vDisk store read / write performance	- (SMB v1) + (SMB v2)	+	+	++
XS – Dom0 overhead		0	0 (soft. Initiator) ++ (hard. Initiator)	++
XS – LUN sizes		++	O	O
XD – Machine Creation Services disk space savings		++	--[1]	--[1]

1: Can be mitigated with array-level thin provisioning

Fault Tolerance

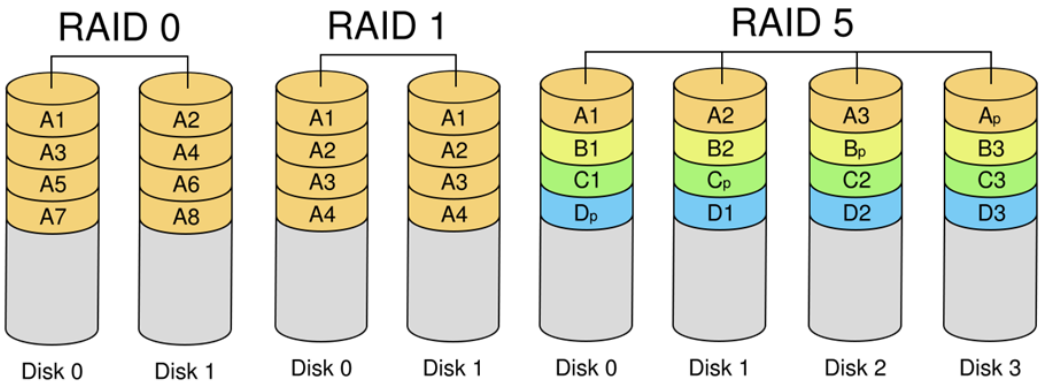
Besides performance data security, which includes access security as well as fault tolerance, is one of the most important aspects within the storage industry. Within this chapter we will discuss common fault tolerance concepts.

Standard RAID Levels

The most basic measure in order to provide fault tolerance is implementing a RAID (Redundant Array of Independent/Inexpensive Disks), which allows "combining multiple disk drive components into a logical unit, where data is distributed across the drives in one of several ways called RAID levels"[20]. The most commonly used RAID levels are:

- **RAID 0:** Striped set no parity. Striping is where each successive block of information is written to alternate disks in the array. RAID 0 still suffers from a single disk failure in the array, but is often used to get the increased read-speed. The increase in read-speed comes from being able to simultaneously move the disk read/write heads for the different drives containing the sequential block to be read. Write speeds may also improve, since each sequential blocks can be written at the same time to the different disks in the array.
- **RAID 1:** Mirroring, no parity. Mirroring is where each block is duplicated across all disks in the array. Here, any one disk failure will not impact data integrity. Better read speeds are achieved by using the drive whose read/write head is closest to the track containing the block to be read. There is generally no improvement in write speeds.
- **RAID 5:** Striped set with distributed parity. The advantage here is that the data from one drive can be rebuilt with the parity information contained on the other drives. RAID 5 can only afford 1 drive to fail.
- **RAID 6:** Striped set similar to RAID 5 but with double distributed parity and the ability to sustain two failed drives.

The following diagram depicts the levels described above:



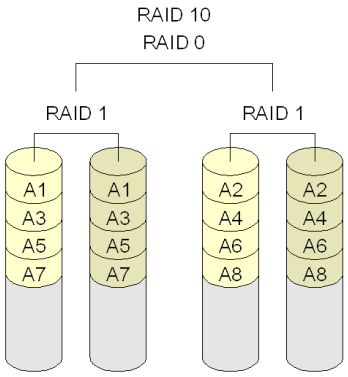
[20] http://en.wikipedia.org/wiki/RAID

Nested RAID Levels

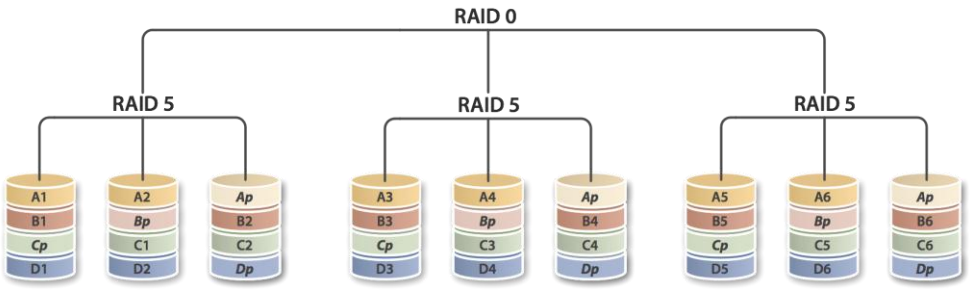
"Levels of nested RAID, also known as hybrid RAID, combine two or more of the standard levels of RAID (redundant array of independent disks) to gain performance, additional redundancy, or both."[21]

Commonly seen nested RAID levels in Citrix environments are:

- **RAID 10:** A RAID 10 as recognized by the storage industry association and as generally implemented by RAID controllers is a RAID 0 array of mirrors (which may be two way or three way mirrors) and requires a minimum of 4 drives. Each disk access is split into full-speed disk accesses to different drives, yielding read and write performance like RAID 0. All but one drive from each RAID 1 set could fail without damaging the data. However, if the failed drive is not replaced, the single working hard drive in the set then becomes a single point of failure for the entire array.



- **RAID 50:** A RAID 50 combines the straight block-level striping of RAID 0 with the distributed parity of RAID 5. It requires at least 6 drives. RAID 50 improves upon the performance of RAID 5 particularly during writes, and provides better fault tolerance than a single RAID level does. One drive from each of the RAID 5 sets could fail without loss of data. However, if the failed drive is not replaced, the remaining drives in that set then become a single point of failure for the entire array. This level is recommended for applications that require high fault tolerance, capacity and random positioning performance.



[21] http://en.wikipedia.org/wiki/Nested_RAID_levels

The following table outlines the key quantitative attributes of the most commonly used RAID levels:

RAID Level	Capacity	Fault Tolerance	Read Performance (random)	Write Performance (random)
0	100%	None	Very Good	Very Good (Write Penalty 0)
1	50%	Good	Very Good	Good (Write Penalty 2)
5	Disk size * (# of disks -1)	Good	Very Good	Bad (Write Penalty 4)
10	50%	Very Good	Very Good	Good (Write Penalty 2)
50	(Disk size * (# of disks -1)) * # of RAID sets	Very Good	Very Good	Good (as striped) (Write Penalty 4)

Multipath I/O

Multipath I/O is a fault-tolerance and performance enhancement technique. "Multipathing solutions use redundant physical path components — adapters, cables, and switches — to create logical paths between the server and the storage device. In the event that one or more of these components fails, causing the path to fail, multipathing logic uses an alternate path for I/O so that applications can still access their data. Each network interface card (in the iSCSI case) or HBA should be connected by using redundant switch infrastructures to provide continued access to storage in the event of a failure in a storage fabric component."[22]

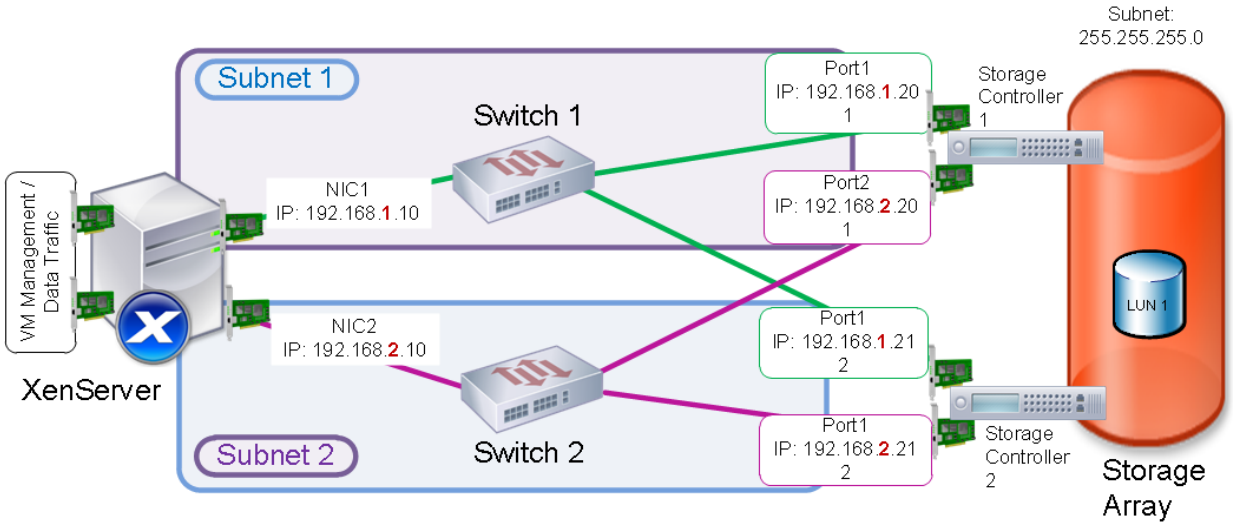
Should one controller, port or switch fail, the server's OS can route I/O through the remaining controller transparently to the application, with no changes visible to the applications, other than perhaps incremental latency.

But, the same logical volume within a storage device (LUN) may be presented multiple times to the server through each of the possible paths. In order to avoid this and make the device easier to administrate and to eliminate confusion, multipathing software is needed. This is responsible for making each LUN visible only once from the application and OS point of view. In addition to this, the multipathing software is also responsible for fail-over recovery, and load balancing:

- **Fail-over recovery:** In a case of the malfunction of a component involved in making the LUN connection, the multipathing software redirects all the data traffic onto other available paths.
- **Load balancing:** The multipathing software is able to balance the data traffic equitably over the available paths from the hosts to the LUNs.

[22] http://technet.microsoft.com/en-us/library/cc725907.aspx

The following diagram outlines a sample HA configuration for an iSCSI based storage attached to a XenServer:



In the environment shown above, XenServer MultiPathing allows storage I/O to leverage multiple redundant network paths to the storage infrastructure, which can increase performance and provide high availability. Once again, the storage array must support this configuration. As previously discussed, this diagram shows a dedicated, isolated switch and network infrastructure for storage traffic.

Storage Replication

"The most basic method is disk mirroring, typical for locally-connected disks. The storage industry narrows the definitions, so *mirroring* is a local (short-distance) operation. A *replication* is extendable across a computer network, so the disks can be located in physically distant locations, and the master-slave database replication model is usually applied. The purpose of replication is to prevent damage from failures or disasters that may occur in one location, or in case such events do occur, improve the ability to recover. For replication, latency is the key factor because it determines either how far apart the sites can be or the type of replication that can be employed.

The main characteristic of such cross-site replication is how write operations are handled:

- **Synchronous replication -** guarantees "zero data loss" by the means of atomic write operation, i.e. write either completes on both sides or not at all. Write is not considered complete until acknowledgement by both local and remote storage. Most applications wait for a write transaction to complete before proceeding with further work, hence overall performance decreases considerably. Inherently, performance drops proportionally to distance, as latency is caused by speed of light. For 10 km distance, the

fastest possible round-trip takes 67 μs, whereas nowadays a whole local cached write completes in about 10-20 μs.

  - An often-overlooked aspect of synchronous replication is the fact that failure of *remote* replica, or even just the *interconnection*, stops by definition any and all writes (freezing the local storage system). This is the behavior that guarantees zero data loss. However, many commercial systems at such potentially dangerous point do not freeze, but just proceed with local writes, losing the desired zero recovery point objective.

- **Asynchronous replication -** write is considered complete as soon as local storage acknowledges it. Remote storage is updated, but probably with a small lag. Performance is greatly increased, but in case of losing a local storage, the remote storage is not guaranteed to have the current copy of data and most recent data may be lost.

- **Semi-synchronous replication -** this usually means that a write is considered complete as soon as local storage acknowledges it and a remote server acknowledges that it has received the write either into memory or to a dedicated log file. The actual remote write is not performed immediately but is performed asynchronously, resulting in better performance than synchronous replication but with increased risk of the remote write failing.

  - Point-in-time replication - introduces periodic snapshots that are replicated instead of primary storage. If the replicated snapshots are pointer-based, then during replication only the changed data is moved not the entire volume. Using this method, replication can occur over smaller, less expensive bandwidth links such as iSCSI or T1 instead of fiber optic lines."[23]

Depending on the details behind how the particular replication works, the application layer may or may not be involved. If blocks are replicated without the knowledge of file systems or applications built on top of the blocks being replicated, when recovering using these blocks, the file system may be in an inconsistent state.

- A "Restartable" recovery implies that the application layer has full knowledge of the replication, and so the replicated blocks that represent the applications are in a consistent state. This means that the application layer (and possibly OS) had a chance to 'quiesce' before the replication cycle.

- A "Recoverable" recovery implies that some extra work needs to be done to the replicated data before it can be useful in a recovery situation.

[23] http://en.wikipedia.org/wiki/Storage_replication#Disk_storage_replication

RPO & RTO

For replication planning, there are two important numbers to consider:

- **Recovery Point Objective (RPO)** describes the acceptable amount of data loss measured in time. For example: Assume that the RPO is 2-hours. If there is a complete replication at 10:00am and the system dies at 11:59am without a new replication, the loss of the data written between 10:00am and 11:59am will **not** be recovered from the replica. This amount of time data has been lost has been deemed acceptable because of the 2 hour RPO. This is the case even if it takes an additional 3 hours to get the site back into production. The production will continue from the point in time of 10:00am. All data in between will have to be manually recovered through other means.

- The **Recovery Time Objective (RTO)** is the duration of time and a service level within which a business process must be restored after a disaster in order to avoid unacceptable consequences associated with a break in business continuity. The RTO attaches to the business process and not the resources required to support the process.

Snapshots

In general a snapshot is not a full copy, since that would take too long im most, but it's a 'freezing' of all the blocks within the selected storage area, making them read-only at that point in time. "Most snapshot implementations are efficient and … the time and I/O needed to create the snapshot does not increase with the size of the data set."[24] Any logical block that needs to be modified after the snapshot, is allocated a new physical block, thus preserving the original snapshot blocks as a backup. Any new blocks are what take up new space, and are allocated for the writes after the snapshot took place. Allocating space in this manner can take substantially less space than taking a whole copy.

Deleting of a snapshot can be done in the background, essentially freeing any blocks that have been updated since the snapshot.

Snapshotting can be implemented in the management tools of the storage array, or built into the OS (such as Microsoft's Volume Snapshot Service – VSS http://en.wikipedia.org/wiki/Volume_Shadow_Copy_Service). As with RAID, the advantage of building this functionality at the block-level is that it can be abstracted from the file systems that are built on top of the blocks. Being at this low level also has a drawback, in that when the snapshot is taken, the file systems (and hence applications) may not be in a consistent state. There is usually a need to 'quiesce' the running machine (virtual or otherwise) before a snapshot is made. This implies that all levels (up to the application) should be aware that they reside on a snapshot-capable system.

[24] http://en.wikipedia.org/wiki/Snapshot_(computer_storage)

Storage Technologies

Storage Caching

Storage caching is used to buffer blocks of data in order to minimize the utilization of disks or storage arrays and to minimize the read / write latency for storage access. The main task of storage cache is to buffering data which has been read (in case it will be read again) or has been written (send write commitment immediately, to allow the system to continue working without waiting for the data actually being written to disk). Especially for write intensive scenarios such as virtual desktops, write caching is very beneficial as it can keep the storage latency even during peak times at a low level.

Storage Cache can be implemented in four places:

- Disk (embedded memory – typically non-expansible)
- Storage Array (vendor specific embedded memory + expansion cards)
- Computer accessing the Storage (RAM)
- Storage Network (i.e. Provisioning Server)

The cache can be subdivided into two categories:

- Volatile Cache: Contained data is lost upon power outages (good for reads or non-critical writes)
- Non-Volatile Cache: Data is kept safe in case of power outages (good for reads and writes). Often referred as Battery Backed Write Cache

To further increase the speed of the disk or storage array advanced algorithms such as Read-ahead / Read-behind or Command Queuing are commonly used.

Thin Provisioning & Over-Allocation[25]

Thin Provisioning, in a shared storage environment, is a method for optimizing utilization of available storage. It relies on on-demand allocation of blocks of data versus the traditional method of allocating all the blocks up front. This methodology eliminates almost all whitespace which helps avoid the poor utilization rates, often as low as 10%, that occur in the traditional storage allocation method where large pools of storage capacity are allocated to individual servers but remain unused (not written to). This traditional model is often called "fat" or "thick" provisioning.

With thin provisioning, storage capacity utilization efficiency can be automatically driven up towards 100% with very little administrative overhead. Organizations can purchase less storage capacity up front, defer storage capacity upgrades in line with actual business usage, and save the operating costs (electricity and floor space) associated with keeping unused disk capacity spinning.

Previous systems generally required large amounts of storage to be physically pre-allocated because of the complexity and impact of growing volume (LUN) space. Thin provisioning enables over-allocation or over-subscription.

Over-allocation or over-subscription is a mechanism that allows a server to view more storage capacity than has been physically reserved on the storage array itself. This allows flexibility in growth of storage volumes, without having to predict accurately how much a volume will grow. Instead, block growth becomes sequential. Physical storage capacity on the array is only dedicated when data is actually written by the application, not when the storage volume is initially allocated. The servers, and by extension the applications that reside on them, view a full size volume from the storage but the storage itself only allocates the blocks of data when they are written. Close monitoring of the physically available storage is vital when using over-allocation, as all systems and applications accessing the particular storage (or LUN) will be affected immediately and not be able to write any further data.

[25] http://en.wikipedia.org/wiki/Thin_provisioning

Data De-Duplication[26]

This is an advanced form of data compression. Data de-duplication software as an appliance, offered separately or as a feature in another storage product, provides file, block, or sub-block-level elimination of duplicate data by storing pointers to a single copy of the data item. This concept is sometimes referred to as data redundancy elimination or single instance store. The effects of de-duplication primarily involve the improved cost structure of disk-based solutions. As a result, businesses may be able to use disks for more of their backup operations and be able to retain data on disks for longer periods of times, enabling restoration from disks.

De-Duplication may occur "in-line", as data is flowing, or "post-process" after it has been written.

- **Post-process de-duplication:** With post-process de-duplication, new data is first stored on the storage device and then a process at a later time will analyze the data looking for duplication. The benefit is that there is no need to wait for the hash calculations and lookup to be completed before storing the data thereby ensuring that store performance is not degraded. Implementations offering policy-based operation can give users the ability to defer optimization on "active" files, or to process files based on type and location. One potential drawback is that you may unnecessarily store duplicate data for a short time which is an issue if the storage system is near full capacity.

- **In-line de-duplication:** This is the process where the de-duplication hash calculations are created on the target device as the data enters the device in real time. If the device spots a block that it already stored on the system it does not store the new block, just references to the existing block. The benefit of in-line de-duplication over post-process de-duplication is that it requires less storage as data is not duplicated initially. On the negative side, it is frequently argued that because hash calculations and lookups takes so long, it can mean that the data ingestion can be slower thereby reducing the backup throughput of the device. However, certain vendors with in-line de-duplication have demonstrated equipment with similar performance to their post-process de-duplication counterparts.

[26] http://en.wikipedia.org/wiki/Data_deduplication

Revision History

Revision	Change Description	Updated By	Date
1.0	Initial Document	Olivier Withoff - Architect	August 27, 2008
2.0	Document Update and Expansion	Thomas Berger – Architect Daniel Feller – Lead Architect Chris Gilbert - Sr Software Dev Engineer Martin Rowan – Director Mark Nijmeijer – Director Tarkan Kocoglu – Director	September 9, 2011

About Citrix

Citrix Systems, Inc. (NASDAQ:CTXS) is the leading provider of virtualization, networking and software as a service technologies for more than 230,000 organizations worldwide. Its Citrix Delivery Center, Citrix Cloud Center (C3) and Citrix Online Services product families radically simplify computing for millions of users, delivering applications as an on-demand service to any user, in any location on any device. Citrix customers include the world's largest Internet companies, 99 percent of Fortune Global 500 enterprises, and hundreds of thousands of small businesses and consumers worldwide. Citrix partners with over 10,000 companies worldwide in more than 100 countries. Founded in 1989, annual revenue in 2010 was $1.9 billion.