

Brought to you by **HITACHI**
Inspire the Next
©Hitachi Data Systems

Storage Virtualization FOR DUMMIES[®]

Hitachi Data Systems Edition

Maximize storage
utilization and
simplify management

**A Reference
for the
Rest of Us!**[®]

FREE eTips at dummies.com[®]



Adrian De Luca
Mandar Bhide

Storage Virtualization in Three Layers of Infrastructure

Modern storage virtualization technologies pool heterogeneous storage vendor products together in a specific way to provide advanced features such as non-disruptive migration of data and thin provisioning (explained in Chapter 5). This level of abstraction can be implemented in three layers of the infrastructure, in the server, in the storage network, and in the storage controller.

In the server

Some of the earliest forms of storage virtualization came not from the storage infrastructure, but from within the server, or, more specifically, the server's operating systems.

With traditional storage hardware devices that connected directly to servers, the actual magnetic disk was presented to servers and their operating systems as LUNs, where the disk was arranged into sectors comprised of a number of fixed-size blocks. To allow applications to not only store, but find information easily, the operating system arranged these blocks into a "file system." Much like a paper-based filing system, a file system is simply a logical way of referencing these blocks into a series of unique files, each with a meaningful name and type so they can be easily accessed. For example, take the file name `My_Summer_Break.doc`. `My_Summer_Break` describes what is contained within the file and the extension `.doc` identifies the file as a document. This naming is a lot more meaningful than just a numeric block number.

Although file systems helped to reference information easily, as more and more of them were created, exhausting the storage space of the physical LUN, another LUN would be created and given to the operating system to continue storing files. To know which data was stored on what LUN, the operating system would assign each one a volume number, name, or identifier. In Microsoft Windows you are most likely familiar with a letter given to each volume such as `C:\` or `D:\`, whereas in UNIX these look like `/dev/hd0` or `/dev/hd1`. As applications and users created more files, more volumes were needed to keep up, pretty soon making it very difficult to manage.

Then operating system vendors came up with the concept of a *logical volume manager (LVM)*. Much like how file systems grouped blocks together to present files, LVMs grouped volumes or LUNs together to present larger, more flexible storage pools to applications, as shown in Figure 2-2. When an LVM started to run out of space, you could *concatenate*, or add another volume, to make it larger without having to reconfigure the application or shut it down. Conversely, if you had a large volume that you wanted to slice into smaller chunks, LVMs would let you partition it in order to separate different information, say one slice for the operating system itself and the other for user data.

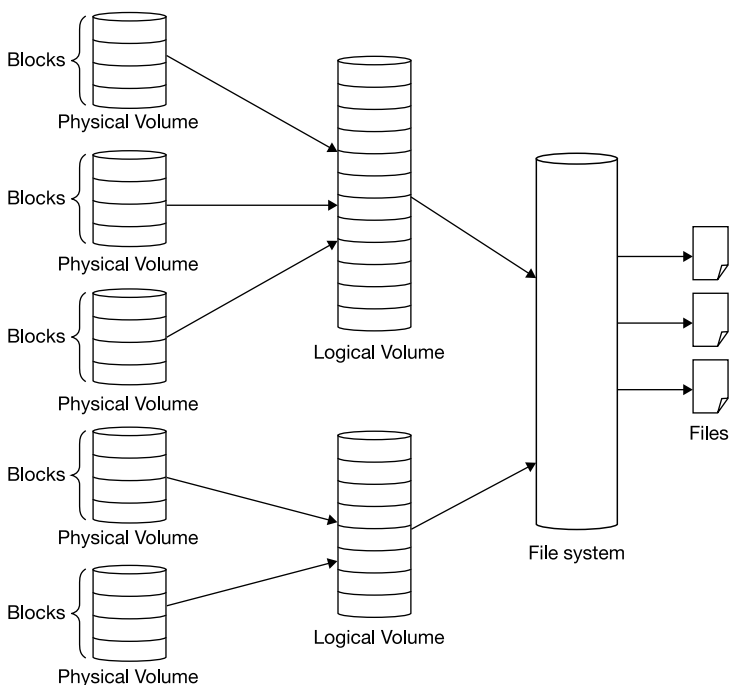


Figure 2-2: Logical volume managers (LVMs) store information in file systems.

Also, since LVMs could hide or abstract where the storage was actually coming from, you could present multiple physical disks and perform a “striping” operation. The technique of *data striping* takes multiple physical disks, ideally with their own discrete path to the server, breaks up the files into even pieces, and spreads the file across multiple storage devices. Writing and reading data in this way reduces the response time and thereby

increases the performance. All hard disks have a *seek time*, which is the time it takes for the information being requested on the spinning disk to arrive under the reading head. By spreading data across multiple disks, the process of finding the information can be done in parallel, thereby reducing the seek time.

Although striping greatly helped to retrieve data faster, it did pose a significant risk that, if just one of the disks failed, then effectively the full data could not be retrieved, resulting in corruption. In the 1980s, a new striping method was introduced, RAID, which increased data reliability if disks failed. RAID-1 duplicates all the data across every disk, providing a backup copy for everything. Although this provides high levels of resilience, it is extremely expensive because you need double the amount of storage. RAID-5, shown in Figure 2-3, combines the performance of striping plus parity, which provides recovery in the event of a disk failure. By using a mathematical formula to calculate the parity, if a disk were to fail in the RAID set, then the data can be regenerated and retrieved with no impact. More recently, RAID-6 was introduced to cope with the long regeneration times of large disk drives, writing parity information to two disks instead of one.

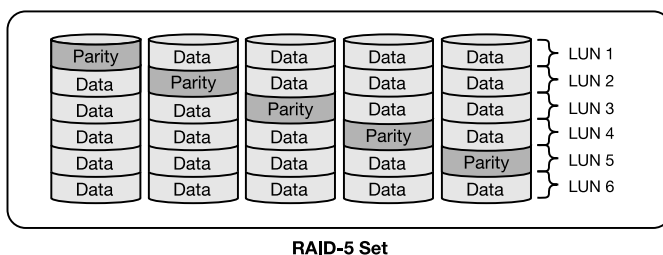


Figure 2-3: How RAID increases resilience to minimize data loss.



Server-based forms of storage virtualization were originally incorporated into operating systems as software and still remain very popular today. Here are the key benefits of this approach:

- ✓ Server-based storage virtualization is highly configurable and flexible since it's implemented in the system software.
- ✓ Because most operating systems incorporate this functionality into their system software, it is very cheap.
- ✓ It does not require additional hardware in the storage infrastructure, and works with any devices that can be seen by the operating system.

A number of downsides to server-based virtualization also exist:

- ✔ Although it helps maximize the efficiency and resilience of storage resources, it's optimized on a per-server basis only.
- ✔ The task of mirroring, striping, and calculating parity requires additional processing, taking valuable CPU and memory resources away from the application.
- ✔ Since every operating system implements file systems and volume management in different ways, organizations with multiple IT vendors need to maintain different skill sets and processes, with higher costs.
- ✔ When it comes to the migration or replication of data (either locally or remotely) it becomes difficult to keep track of data protection across the entire environment.

Most operating system vendors like Microsoft, IBM, Hewlett Packard, and RedHat (Linux) provide at least some capability to virtualize storage resources. Vendors like Symantec offer more advanced forms of server-based storage virtualization.

In the storage network

With the introduction of network attached storage (NAS) and storage area networks (SANs) in the late 1990s, it became possible to separate disks (and their controllers) from servers and share the storage resources more effectively among all applications in the IT environment. The storage network became the traffic cop for all information being exchanged between servers and storage devices, and some storage vendors thought this would be the perfect place to manage virtualization.



Network-based storage virtualization embeds the intelligence of managing the storage resources in the network layer, abstracting the view of real storage resources between the server and the storage array, either in-band or out-of-band.

The *in-band* approach, sometimes referred to as *symmetric*, embeds the virtualization functionality in the I/O (input/output) path between the server and storage array, shown in Figure 2-4, and can be implemented in the SAN switches themselves or in specialized appliances. All I/O requests, along with the data, pass through the device, with the server interacting with the

virtualization device, never directly with the storage device. The virtualization device analyzes the request, consults its mapping tables, and, in turn, performs I/O to the storage device. These devices not only translate storage requests but are also able to cache data with their on-board memory, provide metrics on data usage, manage replication services, orchestrate data migration, and implement thin provisioning.

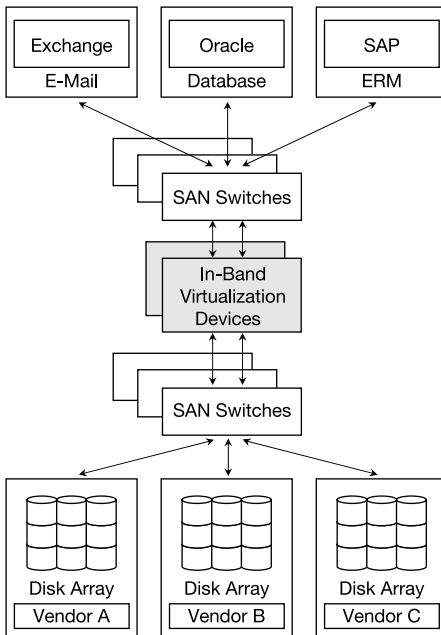


Figure 2-4: In-band network storage virtualization is embedded in the I/O path.

The *out-of-band* approach, sometimes referred to as *asymmetric*, does not strictly reside in the I/O path like the in-band approach; rather, it works hand in hand with specific virtualization-enabled SAN switches to perform specific look-ups, shown in Figure 2-5. The servers maintain direct interaction with the storage array through the intelligent switch. The out-of-band appliance maintains a map (often referred to as *meta-data*) of all the storage resources connected in the SAN and instructs the server where to find it. In this two-step process, the server uses

special software or an agent, as instructions need to be sent through the SAN to make it work. As data never passes through the virtualization device, performance is only slightly impacted; however, functions such as caching of data are not possible.



Both in-band and out-of-band approaches provide storage virtualization with the ability to:

- ✔ Pool heterogeneous vendor storage products in a seamless accessible pool.
- ✔ Perform replication between non-like devices.
- ✔ Provide a single management interface.

However, only the in-band approach can cache data for increased performance.

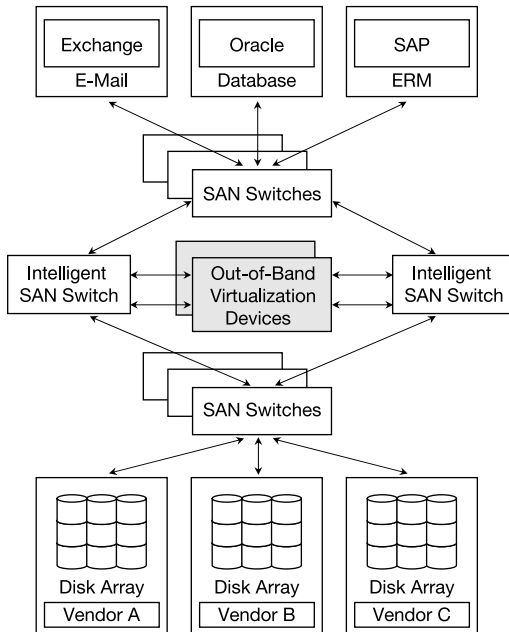


Figure 2-5: Out-of-band network storage virtualization uses intelligent switches to maintain direct interaction between the servers and the storage arrays.

Both approaches also suffer from a number of drawbacks:

- ✔ Implementation can be very complex because the pooling of storage requires the storage extents to be remapped into virtual extents. This requires a mapping table, which becomes a single point of failure, and a vendor lock-in. Migration to a different virtualization solution is extremely difficult or impossible as all the data must be moved back through the mapping table.
- ✔ The virtualization devices are typically servers running system software and requiring as much maintenance as a regular server. Clustering is needed to protect the mapping tables and maintain cache consistency between the nodes, which can be risky. Servers are also limited in the number of storage ports they can support and RAM (random access memory) capacity (as storage is limited in cache). As a result, servers lack scalability.
- ✔ The I/O can suffer from latency, impacting performance and scalability due to the multiple steps required to complete the request, and limited to the amount of memory and CPU available in the appliance nodes.
- ✔ Decoupling the virtualization from the storage once it has been implemented is impossible because all the meta-data resides in the appliance, thereby making it proprietary.
- ✔ Solutions on the market only exist for fibre channel (FC) based SANs. These devices are not suitable for Internet protocol (IP) based SANs, which utilize iSCSI (Internet small computer system interface), NAS, or mainframe servers.
- ✔ Since both approaches are dependent on the SAN, they require additional switch ports, which involves additional zoning complexity.
- ✔ When migrating data between storage systems, the virtualization appliance must read and write the data through the SAN, check status coming back, and maintain a log for any changes during the move that impact performance.
- ✔ All of these add complexity and cost to the SAN, which makes it very difficult to manage the network with *registered state change notifications* (RSCNs, which notify specified nodes of any major fabric changes), inter-switch chatter, zoning changes, and buffer credit management. SANs are the third leading cause of application failure after human and software errors.

