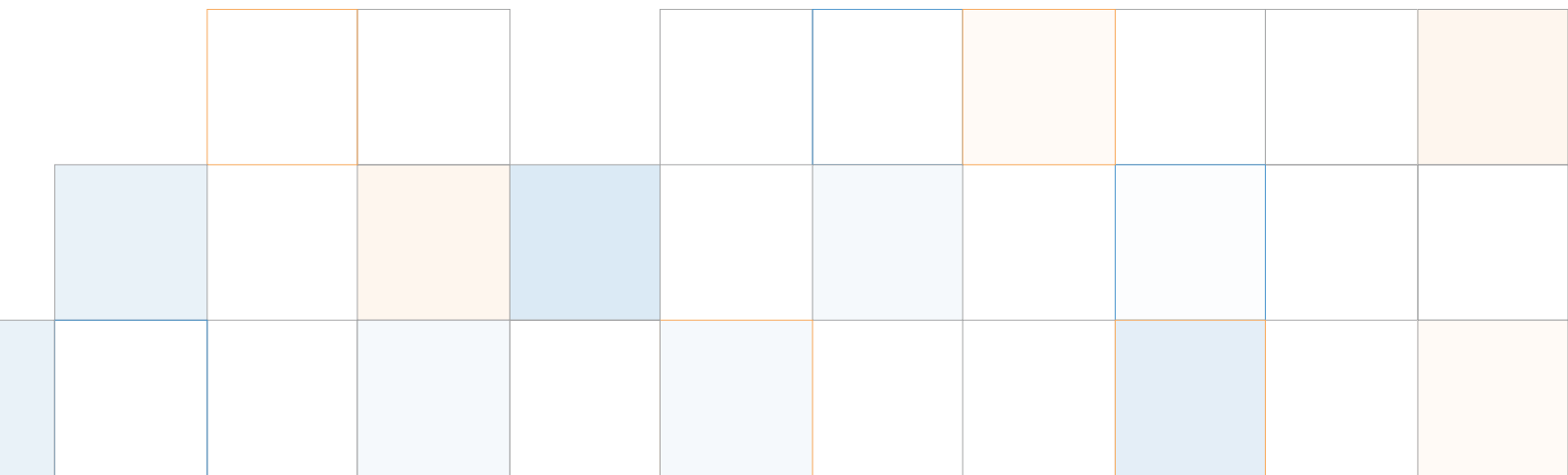




articles

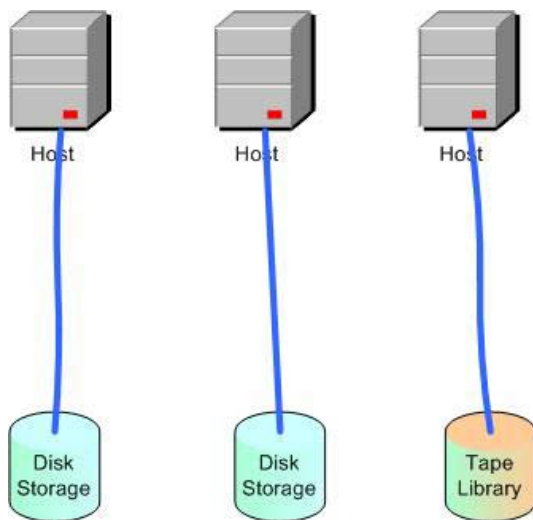


Hardware: SAN

Hardware: SAN

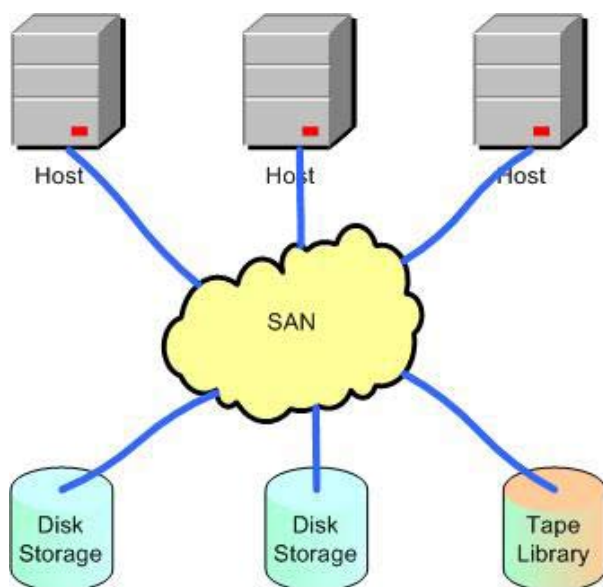
Put simply, a Storage Area Network (SAN) is a network whose primary purpose is to facilitate the data transfer between computer systems (hosts) and storage devices. This storage device is normally either a disk array or tape drive/library. Generally, a SAN consists of fibre channel based communication. The method of signalling and communication is different to shared network storage, as a SAN facilitates direct block I/O requests between a host and storage, much like locally attached SCSI devices, whereas when accessing shared network storage, requests are sent based on file requests, over CIFS, NFS, etc.

Traditional storage provided for normally one or two hosts to be able to access each device in a shared cabling environment, although device access was limited to one host at a time. This was primarily via SCSI bus, although other methods were widely used, such as SSA, IDE, etc.



SANs allow multiple hosts to connect simultaneously to multiple storage devices (through means of storage partitioning), facilitating High Availability (HA) storage configurations. These connections are routed through a Fabric Switch.

On a SAN, each host can access any storage on the SAN, as long as it is defined on the switch zoning configuration.



FC Zoning

Firstly, Zoning can be likened to VLANs on an ethernet switch. It is the access control mechanism governing the flow of traffic between two or more switchports on the FC switch. To identify each FC switch as a separate FC path, each FC switch must have a different domain ID though.

The basics:

Each switchport can be assigned an Alias. The switchport on the Brocade switches is known by the location code containing the Domain ID and port number, in the form of “,“.... eg “1,3 . An alias is string which normally describes the port. For example, the alias for “1,3 could be “FAStT2_HBA1 . An alias is referred later as a “member”, when defining zones.

A Zone is the mapping between the ports, and to define the zone, you can use aliases or port numbers. Aliases will probably save confusion later. Each mapping is created as a separate zone, for example, you could define 12 “zones” on the switch. A zone can contain 2 or more members (eg a zone could map port “1,3; 1,4; 1,7). A member can be part of more than 1 zone. A zone is referred later as a “member”, when defining configurations.

A configuration is a group of zones. the configuration on a switch at Sonic Health is called “A1B1 , and contains 4 zones. A configuration can be easily enabled and disabled on the fly. configurations can be updated on the fly, although if you remove a zone which is in use, the host using that zone to connect to storage, etc, will no longer be able to communicate across that zone. (ie be careful).

Configuration Example

This is an example of adding a FAStT to a switch, and then creating some zones to allow the AIX host to connect to the FAStT.

telnet into the switch

There are 4 users on a switch. Normally “admin” will allow you to do everything that you need. The other users are “root”, “factory”, and “user”.

the command “help” will assist if you need further assistance

the command “zonehelp” will provide specific zoning assistance.

Let us assume that the following devices are configured on the following ports:

lurch (AIX server 1) - 1,0 morticia (AIX server 2) - 1,1 FAStT 1 (Storage) - 1,2

ie, the domain ID is 1, the ports are 0 through to 3

We wish to add “FAStT 2 (storage) to port 1,3. After connecting it physically, the

configuration needs to be

lurch (AIX server 1) - 1,0 morticia (AIX server 2) - 1,1 FAStT 1 (Storage) - 1,2 FAStT 2 (Storage) - 1,3

The current configuration is displayed through the command

`alishow`

```
TRTSWITCH:root> alishow
Defined configuration:
cfg:   trt   lurch1; morticial
zone:  lurch1   FAStT1_hba1;lurch_hba1
zone:  morticial   FAStT1_hba1;morticia_hba1
alias: lurch_hba1
1,0
alias: morticia_hba1
1,1
alias: FAStT1_HBA1
1,2

Type to continue, Q to stop:
Effective configuration:
cfg:   A1B1
zone:  lurch1   1,0
      1,2
zone:  morticial 1,1
      1,2
value = 0
```

```

TRTSWITCH:root> alicreate
Usage: aliCreate aliName, aliMemberList
value = -1
TRTSWITCH:root> alicreate "FASTt2_HBA1", "1,3"
value = 0
TRTSWITCH:root> zonecreate "lurch2", "FASTt2_HBA1; lurch_hba1"
value = 0
TRTSWITCH:root> zonecreate "morticia2", "FASTt2_HBA1; morticia_hba1"
value = 0
TRTSWITCH:root> cfgadd
Usage: cfgAdd cfgName, cfgMemberList
value = -1
TRTSWITCH:root> cfgadd "trt", "lurch2; morticia2"
value = 0
TRTSWITCH:root> cfgsave
Starting the Commit operation...
cfgSave successfully completed
value = 0
TRTSWITCH:root> cfgenable "trt"
Usage: cfgEnable config
value = 0
TRTSWITCH:root> alishow
Defined configuration:
cfg:  trt    lurch1; morticial1;lurch2;morticia2
zone:  lurch1    FASTt1_hba1;lurch_hba1
zone:  morticial1    FASTt2_hba1;morticia_hba1
zone:  lurch2    FASTt1_hba1;lurch_hba1
zone:  morticia2    FASTt2_hba1;morticia_hba1
alias: lurch_hba1
1,0
alias: morticia_hba1
1,1
alias: FASTt1_HBA1
1,2
alias: FASTt2_HBA1
1,3

```

```

Type to continue, Q to stop:
Effective configuration:
cfg:  A1B1
zone:  lurch1      1,0
      1,2
     zone:  morticial 1,1
      1,2
zone:  lurch2      1,0
      1,3
zone:  morticia2   1,1
      1,3
value = 0

```

You can adapt this for creating new zones, removing configurations, etc.

LUNs

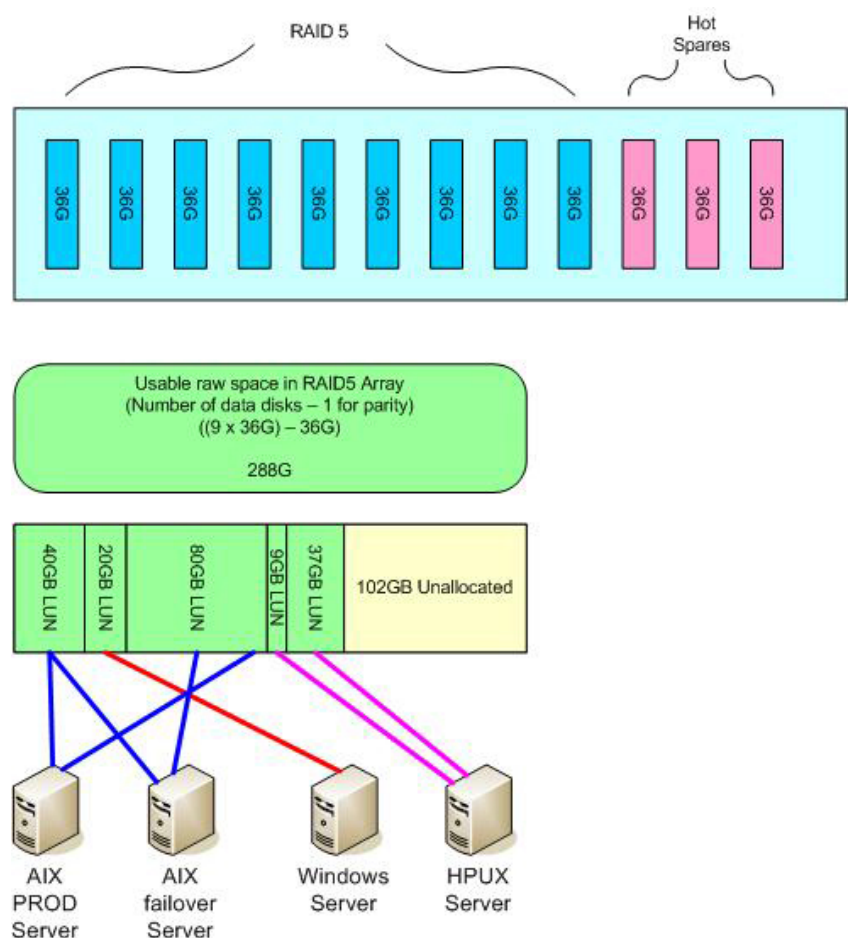
SAN Storage presents itself to hosts in the form of Logical UNits (LUNs). Luns are a versatile approach to dividing up storage, allowing it to be used in exactly the right proportions for the desired host systems.

Traditional storage was not easily divided up this way. If a RAID5 array was 288GB, all of that RAID5 array would be presented to only 1 host, and appear as 1 massive volume. If only 50GB was required, this would mean that 238GB would be “wasted”.

To address this issue, SANs are divided logically into LUNs. A RAID array can be divided up into portions of any desired size, allowing it to meet exact requirements on each host, without wasting space. As LUNs are logical divisions of a RAID array, they do not relate at all to physical locations of data on physical disk spindles.

Each LUN can then be allocated to one or more hosts. LUNs are still raw block devices, so a SAN Storage Array can be divided up into LUNs for AIX, HPUX, Windows, Solaris, and many other host types. Each LUN can be allocated to more than one host, allowing for High Availability configurations.

Consider the diagram:



The storage system physically contains 9 x 36GB FC disks in the RAID5 Array. Parity is striped across all of the disks, and in total, takes up 36GB of space. This means that for 9 physical disks worth of space, there are 8 physical disks worth of “usable” space.

The RAID array is one large 288GB block I/O area of storage. To make this storage usable to different hosts, it is divided up logically, into specific sizes. Normally this is configurable, down to the exact “megabytes”, which means no space needs to be unnecessarily wasted. Each division of space is called a LUN.

The storage system is then configured to specify a “host type” for each LUN, and then allocate each LUN to specific hosts. The 40GB LUN and the 80GB LUN is allocated to the “AIX PROD Server” and the “AIX Failover Server”. Although each LUN can only be used by one host at any one time, if the first host failed, and the failover host needed to recover from the failure, it can immediately use the same LUNs as the original host was using.

There is still 102GB of unallocated storage space. Additional LUNs can be configured at any time and allocated to the host and used by the host on the fly. This depends on the host’s capabilities (supported under AIX but not necessarily for Windows, etc)

As well as connecting AIX hosts, the same storage subsystem has been configured to present storage to a HPUX host, and a Windows host.

