# Stretched Clusters

# Eliminating the Need for Disaster Recovery

## Defining the Need

Clustering servers for HA (High Availability) has been with us for several decades. The basic concepts have not changed during that time but the tools for implementing clusters have matured considerably. The one basic characteristic of clusters that is only now being changed by the IT industry is the physical geographic footprint of the cluster.

The basic purpose of the cluster is to eliminate any SPOF (Single Point Of Failure) of any functional component composing the cluster, to build into the cluster redundancy of all systems. Some examples of these are: boot disks, network interfaces, CPUs, data disks, access channels to data disks (NAS, SAN, SCSI, etc.), power, and internet access. Until recently the biggest SPOF for almost all clusters was the rack or certainly the data center that the cluster was hosted within. I have witnessed an entire three noded cluster brought down by a technician accidentally removing power to one rack in a data center, this rack housed ALL components of the HA cluster.

System Administrators have been proactively moving various components of any cluster to non-contiguous locations within a data center. This is the first step is eliminating larger and larger failure footprints from impacting the functions of the HA cluster.

## The Idea

The idea was put forth several years ago that any disaster has a specific and readily measurable “disaster footprint”. This is defined as the area that the disaster impacts. The same concept can be used for an HA cluster. It too can have a measurable geographic footprint, a footprint that until very recently has been measured in just a few square feet.

DR (Disaster Recovery) planning has been a hotly discussed topic for anyone running a data center, along with its close colleague BC (Business Continuance). For this article I will define a DR plan as: “Once the disaster occurs what needs to happen to get to our previous capacity, capabilities and functionality”. The respective definition for a BC plan is: “What do we do in the time interval between the disaster and the completion of the DR plan.”

If we use the various definitions posited above and scale them up we can see where all of this is going. It builds on our current implementations of the cluster architecture to eliminate SPOFs by distributing hardware in non-continuous locations in the data center. This expands the footprint of the cluster to many hundreds of square feet. It eliminates the “clumsy technician” accident from bringing down the cluster. This is due to the fact that the clumsy technician accident has a disaster footprint of less then 10 square feet (in most cases). Thus the cluster is no longer susceptible to the clumsy technician accident because the cluster footprint is larger that the disaster footprint.

In my work with various EOCs (Emergency Operations Centers) each site has a uniquely defined MCA (Maximum Credible Accident). This is defined as the largest, highest impacting event that the site prepares for handling. Each of these MCAs has a measurable footprint, but none of them is as large, or larger, than the geographic footprint that the emergency operations center supervises. If the disaster footprint is larger than the facility footprint it oversees then it is truly a disaster and a Disaster Recovery plan is implemented. This same scenario applies to clusters. If the disaster impacts 100% or more of the cluster the DR and BC plans must be implemented. But my stretching, or enlarging, the cluster footprint to be larger than any credible disaster you eliminate the need for DR or BC planning.

Some data gathered through a local insurance company provided some representative minimum and maximum disaster footprints in table 1.

Table 1

| Disaster | Minimum Footprint | Maximum Footprint |
|---|---|---|
| Clumsy Technician | 1 square foot | Entire building (20,000 sq ft) |
| PDU failure | 1000 sq ft | 10,000 sq ft |
| HVAC failure | 1000 sq ft | 5,000 sq ft |
| Architectural failure | 25 sq ft | 20,000 sq ft |
| Building fire | 10,000 sq ft | 1 sq mile |
| Tornado | 5 sq miles | 50 sq miles |
| Hurricane | 5,000 sq miles | 50,000 sq miles |
| Flood | 5,000 sq miles | 50,000 sq miles |
| Tsunami | 2,000 sq miles | 25,000 sq miles |
| Earthquake | 100 sq miles | 10,000 sq miles |
| Power grid collapse/Electrical failure | 1 sq miles | 500,000 sq miles |

We could talk about huge meteor collisions, gigantic solar flares, orbital change etc. but at that point the disaster is no longer credible, and the ability for anyone to use your protected data will cease.

The overall strategy is to have part of your HA cluster outside of the disaster footprint to continue to provide uninterrupted service before, during, and after a disaster occurs.

## The Architecture

I have currently explored two production clusters that are being run over a MAN (Municipal Area Network). One of these is a cluster with three locations with maximum separation of 10 miles (about 16 km). This cluster is protected from a vast number of disasters, including a 9/11 scenario, building fire or other localized event. One of the themes that was brought up by both groups running these clusters is that the larger the number of sites the smaller the loss of cluster capacity if a single site is affected by a disaster. Counter to this idea is the management issues of a cluster with more then three sites. This increased management load seems to nullify the advantages of say 10 sites with each site having approximately 10% of the clusters capacity.

For this phase of this project project I decided to keep to two sites each with about the same computational capacity and storage capacity. Losing a site would mean losing 50% of my capacity and thus running the cluster in a degraded mode until the lost capacity could be replaced by some means. Running in a degraded mode, while not entirely acceptable, is much better than losing the all ability to process data.

Most SMBs are tending to host data at a facility that specializes in hosting compute hardware. Even larger companies feel that using a hosting facility is a solution that will allow them to concentrate on their business and not have to maintain a large investment in IT departments, class A data center space, maintenance contracts for multiple ISPs, a UPS with generators and fuel contracts, etc. For this project I adopted the same solution for those reason and several others that directly impacted this project. The biggest reason was communications infrastructure: by using a hosting, or colocation, company with at two locations within the US over 500 miles apart I could take advantage of their installed communications infrastructure and backbone for the various communications channels that would be needed.

I chose a company with colocation data centers in the Denver, CO area and Minneapolis, MN. The distance between these facilities is about 680 miles (1120 km), according to United Airlines, and thus met the distance criteria. I worked with the hosting company to provide three separate links accessible to my networks at the two sites. Two of these links were standard Ethernet and one for the SAN network connectivity. I would have preferred to have one more Ethernet and one more SAN link but when you are building on a shoestring and begging resources you take what you can get. I used the two Ethernet links as a production network and an admin network for administrative access, backups etc. I gave up the redundant production network as this was not a true production environment. I also gave up a redundant SAN link and used a single SAN connection between the sites. The admin network also served as one of the two heartbeat links for the VCS cluster, the other heartbeat was done via disk heartbeats, as much as I dislike disk heartbeats it was easier than begging another Ethernet link.

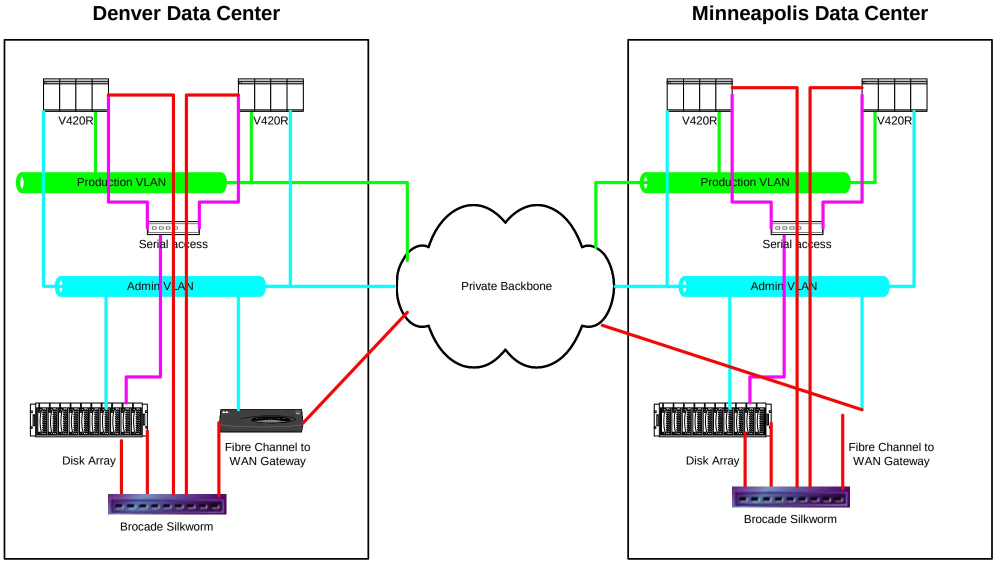
I used two Sun V420R servers at each site running Solaris 9, Oracle 9i and VCS Database Edition for Oracle HA V3.5. Each site also contained a Brocade switch, Cisco switch, FC to WAN gateway, Ethernet to serial adaptor and SAN disk array. The low cost disk array I used was only capable of presenting RAID0 or RAID5 volumes, but this was acceptable for this phase of the project since all mirroring was done at the server level between each site.

The Cisco switches were each equipped with a WIC (WAN Interface Card) to allow VLAN definition to stretch between the clusters on the WAN. This allowed the systems are each site to reside on the same logical VLAN, this would become essential later for BGP routing.

Having worked with Fibre Channel to WAN gateways in Malaysia to provide FC to WAN/ATM access to a remote disk farm I chose to use the same proven technology for this project. By using a FC to WAN gateway I was able to eliminate repeaters and other hardware such as DWDM (Dense Wave Division Multiplexing) at any point between the two sites. This vastly simplified the implementation and also allowed all components critical to the project to be located in the controlled environment of the class A data center.

Once these links were established I had basic network and FC connectivity. With this completed I was able to configure both disk arrays from a single site, set up the volumes, define the aliases and zoning on the Brocade switches and define file systems on mirrored volumes on all systems on the SAN. The overall topology of the stretched cluster is shown in the drawing “Stretched Cluster Topology” that indicated the logical connectivity of each sites major components.

## Stretched Cluster Topology



When I connected the sites to the real world (read: evil internet) I had intended to implement two ISPs for each site to give truly redundant internet access for each site. Again I was only granted access to a single ISP for each site but from a different provider. The original object was to provide redundant internet access for one site and duplicate that access at the other site. Then use BGP routing to give truly redundant access for either site independently. I was able to implement the cluster and still used BGP routing between the two sites allowing independent access to any member of the cluster from either ISP.

The cluster was configured using four service groups running independently on any of the four Sun servers. Failover between the sites worked as expected. It took slightly longer to failover due to the increased latency but the increase was not substantial. Network latency between the sites averaged just over 300 ms during peak daytime hours.

Testing the cluster access was done very simply. Running a script from my notebook connected to the internet via my cell phone I was able to access a simple application and database on a server in the Denver data center. The script simply ran application queries and then the application ran database queries, this simulated a common three tier architecture. Once the access had begun I simply removed the power from the Cisco and FC gateway devices. This instantly stopped all Ethernet communications to all servers in the Denver data center from each other and the ISP that served the Denver data center. It also stopped all FC traffic on the SAN from the Denver site to the Minneapolis site. The application and database service group failovers occurred within 60 seconds and the script again was accessing the application and database. This short failover interval outage was viewed as a success for this phase of the project, although the eventual goal of this project is to eliminate even the fail over outage it seemed like a good first iteration. This cluster can now withstand Denver or Minneapolis falling off the internet map, and people accessing the cluster in another part of the world will have little knowledge about it.

## Challenges

### *Latency*

The latency for both the network and FC connections was the major concern going into this project. They turned out to be of minor significance to the implementation of this cluster. Some of the configuration issues surrounding the increased latency are performance concerns with disk fast writes, database writes and mirroring of volume between data centers.

When performing database writes I tested the performance using fast writes and confirmed writes. With confirmed writes the performance was measurably slower but data validity was ensured. On several of the failover tests using fast writes there was some data loss or corruption concerning writes that were reported to the application as succeeding when they had not been written to disk. I favor implementing the reduced performance to get solid data integrity.

I also considered using a volume replication solution in place of real-time mirroring of data but this solution made the failover a more labor intensive a much less automated task. If Minneapolis falls off the internet map at 02:00 I really would rather sleep through a failover then be paged and have to perform part of the failover manually.

The BGP routing implementation was a little bit touchy to get set up, configured and working correctly. The sheer size of the tables represents a considerable amount of traffic between the Cisco switches before the routes stabilize, on more then one occasion starting the routing from scratch I received some time outs when attempting to synchronize the tables between the sites. But once the tables were synchronized and everything was working it simply continued to work without incident (or until I stuck my fingers into it and caused problems)

### *VLAN stretching*

The logical definition of VLANs across the WAN proved to not be an issue. Trunking of this type between switches has been common for many years; the maturity of the protocol used showed when it tolerated the latency between the sites without complaint. Once set up I only needed to backup the configuration in case I again managed to delete the configuration data.

### *SAN stretching*

Running the SAN over the WAN link was the foundation of this project and caused the highest Rolaids factor. The gateway equipment is not as mature as I would like for this purpose but after several bouts with the manual and customer support I was able to establish connectivity between the data centers. The gateways are intended for high latency connections but the distance I was using them at exceeded what is usually implemented. Thus there was some tuning that needed to be done to keep the link up and stable.

The technical support staff indicated that the latency on the ATM link should never be a concern and that future systems would be shipped with the tuning values that we modified to get a solid connection.

## Next Steps for the Project

This report only covers the first phase of this project. The goal is to implement a true active/active cluster over a WAN distance with a geographic footprint larger than our MCA. Once this is complete the “lessons learned” can be circulated in the IT community and further developed and matured until these techniques become a common practice. My first production UNIX cluster was a two noded active/passive cluster with a shared SCSI bus for storage. The architecture was fairly new, raw and I had no commercial software to build upon. Now such a cluster is common and simple to implement with support from at least 4 different commercial software packages. It is my hope that WAN based clusters follow the same development path.

If WAN based clusters become more common these clusters will supplant DR and BC. WAN based clusters are by definition disaster tolerant and eliminate the need for disaster recovery plans and business continuance plans. Also eliminating the need to maintain and test these plans. My feeling is that infrequently exercised procedures are the most prone to failure, thus fail at the time they are most needed; in this case following a disaster. Running a cluster in a disaster tolerant mode allows daily operation that can be learned from, tuned, tweaked and improved through iteration. This is the best method to mature a technology.

Infrequently used procedures do not benefit from this type of iteration and incremental improvement. This can be seen in the vast array of approaches to disaster recovery and business continuance. There are no commonly accepted methods of implementing these procedures simply because IT personnel have not had the opportunity to work with them, learn from them and improve them. Only from continual operation and incremental improvement can we hope to come to a consensus as an industry on the best procedures and practices for avoiding the impact of disasters on our IT resources.

To reach the goal of true disaster tolerant clusters the next phase of this project will investigate and implement a solution to one of two prominent issues with the current implementation. I will either tackle eliminating the failover interval or the loss of 50% of the computing capacity when a data center is lost. My current ideas on solving these solutions are as follows:

To eliminate the failover interval the application or database will be required to run in an active/active mode on servers in each data center. My first idea on this would be to attempt to implement Oracle RAC running between the two sites. Having worked for Oracle in a previous life I am familiar enough with the software that I feel I can attempt this even though I am not a DBA. This architecture should also improve the database performance as I can possibly set up Oracle to perform write splitting to handle to the individual mirrors eliminating the need to synchronize the mirrors at the volume management layer.

To eliminate the loss of compute capacity of the cluster when a site is lost I have entertained the idea of using a grid computing model at each site to allow capacity to be quickly scaled up. This is an area I have wanted to work in for some time. All the literature I have read concerning grid computing tells a very appealing tail and I have wondered about what the issues and Rolaids factors involved are with implementation and use.

Other improvements I am considering for future phases of this project are how to best implement backup, restore and vaulting. I have not given much thought to the logical flow of data at each site and how to best protect the data for recovery independent of which site or sites are left standing. The trade off in performing multi-site backups is performing the minimum amount of data backups to keep the backup window short but not sacrificing data recoverability. For without the recovery ability the backups are less then useless. Coupled to this is the vaulting function, again I need to consider the logical data flow for each site and between sites to keep the traffic to a minimum but again without sacrificing data recoverability not only for disasters but for the legal, business and tax requirements. Having done many custom vaulting implementations and developed many scripts, utilities and tools that provide functionality beyond the currently available commercial software. I have not previously encountered a vaulting environment that presents the issues that WAN based clusters present.

## Bio

John Vossler is the Chief Architect for IT Infrastructures, Inc. in the Denver & Colorado Springs area specializing in UNIX/Linux infrastructure computing and storage solutions, with heavy emphasis on clustering, backup & restore and storage administration. IT Infrastructures has been running the Solaris Ready Test Facility for Sun Microsystems, Inc. for two years. This affords us ample opportunity for testing Solaris based software and third party hardware in a Solaris environment. John has been working in the IT arena since the PDP-8 and mainframe were the primary tools and he continues today with Linux, Solaris, AIX and HP-UX, implementing his first production UNIX cluster in 1994. John is proud to say that he never touched a DOS or microsoft system until 1998. John can be reached at John.Vossler@ITInfrastructures.net