Using OpenVMS Clusters for Disaster Tolerance

Keith Parris
System/Software Engineer
HP Services – Systems Engineering

Abstract

This article provides a theoretical discussion of disaster tolerance and disaster recovery as well as detailed, practical guidance for planning, configuring, and managing a disaster-tolerant OpenVMS Cluster system. Disaster tolerance is contrasted with disaster recovery and generously illustrated with many real-world examples. Next, guidelines are presented for selecting appropriate technologies and tools and for selecting appropriate sites for multi-site configurations. Then, the management and monitoring tools that are available for OpenVMS disaster tolerant configurations are discussed. This article concludes with an in-depth discussion of important tasks and techniques for managing OpenVMS multi-site disaster tolerant systems and with examples that illustrate the value of OpenVMS Cluster disaster-tolerant configurations.

Terminology

First, the definitions of some basic terms are important to understanding how disaster tolerance fits into the spectrum of available solutions for protecting data and for avoiding interruptions to computing operations.

High Availability

High availability (often abbreviated HA) is a term describing the ability for application processing to continue with high probability in the face of common (mostly hardware) failures. Typical technologies used to provide high availability include:

- Redundant power supplies and fans
- RAID for disks
- Clusters of servers
- Multiple network interface cards (NICs); redundant routers
- Facilities which include dual power feeds, n+1 air conditioning units, Uninterruptible power supply (UPS) units, or a generator

Fault Tolerance

Fault tolerance (often abbreviated FT) is a term describing the ability for a computer system to continue operating despite hardware and/or software failures. Technology required to achieve fault tolerance typically includes:

- Special hardware with full redundancy, error-checking, and hot-swap support
- Special software

Fault tolerance provides the highest level of high availability possible within a single datacenter.

Disaster Recovery

Disaster recovery (often abbreviated DR) is a term describing the ability to resume operations after a disaster. A disaster could be as bad as the destruction of an entire datacenter site and everything in it. Since businesses operate based on data, achieving a successful recovery from a disaster implies that off-site data storage of some sort is in place, so that the business can retrieve its operational data and continue doing business despite the destruction of the primary copy of its business data.

Typically, in disaster recovery scenarios, there is some delay before operations can continue. This delay may be many hours long -- possibly days. It is also likely that some business transaction data may have been lost from IT systems in conjunction with the disaster, and that transaction data must be re-entered, perhaps by re-entering it from paper documents.

A successful disaster recovery hinges on the ability to restore, replace, or re-create:

- Business data (and any external business data feeds)
- Facilities
- Systems
- Networks
- User access

Common DR methods include:

- Tape backups, stored offsite. The data on these tapes can be restored to a compatible system after a disaster destroys the systems and data at the primary site.
- Use of a disaster recovery site. A company can contract for the right to use datacenter space and equipment at the facilities of a vendor such as SunGard. When a disaster is declared, a suitable equipment configuration is configured and made available for use. Alternatively, cooperative agreements may be made between companies to share space and equipment in the event of a disaster affecting one company or the other.
- Data vaulting. Here, data is copied, either on a periodic or continuous basis, to mass storage at another site. This saves the delay of restoring from tape after a disaster.
- Hot site. A site fully prepared to handle processing is kept ready for use. This may be used in conjunction with data vaulting so that a copy of the data already resides at the hot site in the event of a disaster. The hot site and the equipment in it may be owned by the company itself, or may be contractedfrom a vendor.

Disaster Tolerance vs. Disaster Recovery

Disaster recovery is the ability to *resume* operations after a disaster. Disaster tolerance (DT) is the ability to *continue operations uninterrupted* despite a disaster. Ideally, disaster tolerance allows one to continue operations uninterrupted despite a disaster:

- Without any appreciable delays
- Without any lost transaction data.

Businesses vary in their requirements with respect to what is an acceptable recovery time after a disaster, and what is an acceptable amount of data loss in conjunction with a disaster. Platforms and technologies also vary widely in their ability to achieve the disaster tolerance ideals of *no data loss* and *zero recovery time*. OpenVMS disaster-tolerant cluster technology today can be used to achieve zero data loss in connection with a disaster, and recovery times in the single-digit seconds range.

Measuring and Quantifying Business Needs for DR / DT

It is important to determine the requirements for disaster recovery and/or disaster tolerance based on the *business needs* first. Then one can determine the set of acceptable technologies that can meet the DR/DT needs of the business in the most cost-effective way.

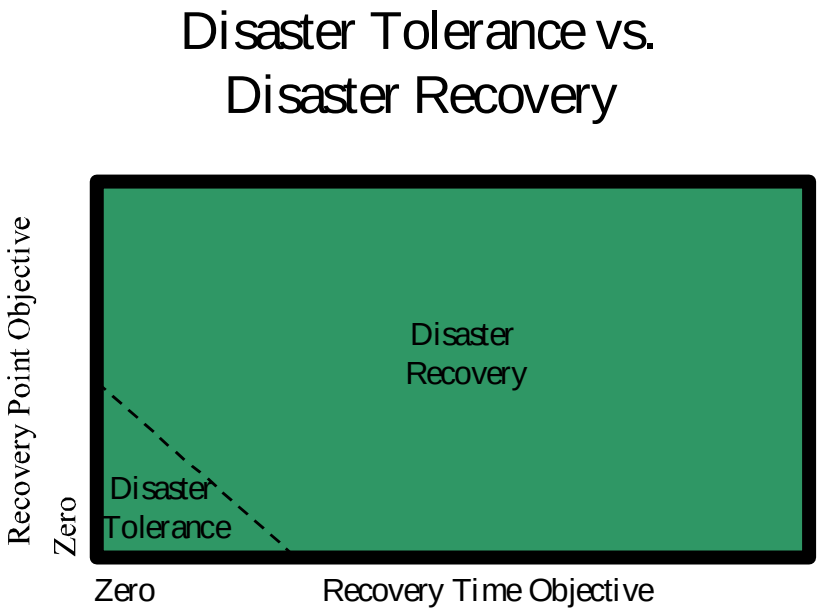
There are a couple of metrics commonly used in the industry today to describe business needs with respect to DR and DT. These are the *recovery point objective* (RPO) and the *recovery time objective* (RTO).

The RPO is used to quantify the amount of data loss that is acceptable, if any. The RTO is used to quantify the amount of downtime that is acceptable, if any. Both are stated in units of time.

For the RPO, the value specified represents the point in time to which one must be able to restore the data in the event of a disaster, and this time is measured relative to the time of the disaster. An RPO of zero represents data current as of the exact time of the disaster, or zero data loss. An RPO of 24 hours means up to one day's worth of transactions might be lost in conjunction with the disaster.

For the RTO, the value is the time duration until business can resume. An RTO of zero represents no downtime at all. An RPO of 24 hours means that one day could elapse before the business could continue its operations.

The main difference, then, between disaster recovery in general and disaster tolerance in particular is that disaster tolerance has the goal of minimizing RPO and RTO.



Let's look at some examples of RPO and RTO values for some real businesses.

I once spoke with a large greeting card manufacturer. Their business requirements were such that they could never, ever lose an order from one of their stores, but since the stores tended to order only once a week or so, downtime of even as much as three days would be acceptable, since a store that could not place an order on one day would be willing to call back a couple of days later and place the order then. But once that order was placed, it absolutely couldn't be lost. These business needs would be described with an RPO of zero and an RTO of 3 days.

I once worked with an online stock brokerage. There, an outage could result in losses of millions of dollars per hour. When customers placed an order and an outage occurred, *and* the price of the stock or option that was the subject of that order changed before the order could be executed, the brokerage had to cover any loss the customers incurred. Conversely, if the customer benefited from the delay, because, for example, they placed an order to sell stock but the outage delayed execution of their order while the price increased, the brokerage did not benefit from the situation; only the customer did. The brokerage suffered all the risk from price changes, and none of the benefits, and the risk increased as the length of an outage increased. Customers who were heavy traders watched the systems closely; any outage longer than just a few seconds would be noticeable to them.

In addition to the issues related to cost of downtime, they could absolutely not afford to ever lose any stock transactions, because of the risk of fraud. If they did lose any transactions in conjunction with an outage, a customer could claim to have sold a large amount of stock just before the outage. If the price of that stock had gone down before the order could be processed, the brokerage would have to cover the difference in price, because there would be no way for them to prove that the order had not actually been

placed. The needs of this business could be described with an RPO of zero and an RTO of perhaps 20 seconds.

An ATM machine makes another interesting example. One might think that because money is involved, zero data loss would be a requirement, but in practice, the requirements are not quite so strict, for two reasons:

1. The monetary losses are fairly limited, because a given customer is only allowed to take out a few hundred dollars at a time
2. The ATM machine itself keeps a log of transactions; if a small number of transactions were lost, it is usually possible to recover the missing data from the ATM log.

Downtime due to an ATM not working can cause losses in terms of lost transaction fees and customer dissatisfaction, but the financial risks of downtime are not large. For this type of business, one might describe the business needs with an RPO of a few minutes, and an RTO of a few minutes as well.

One must be careful, though, because apparently while an ATM machine logs all cash transactions, the log may not include account transfer data. I heard from one gentleman who did a large transfer between accounts using an ATM. The transaction was lost, and lots of large checks started to bounce. Because he had retained the printed receipt of the ATM transaction, he was able to get the fees refunded and receive a sincere apology from the bank.

Also, we know of the case of the Municipal Credit Union (MCU) in Manhattan (see http://www.nwfusion.com/news/2002/0902lessonsside1.html). As a result of the damage of 9/11, their systems were down, but in light of the terrible situation there and out of the goodness of their hearts, they chose to allow their customers to make withdrawals from ATM machines anyway. As a result, 4,000 people bilked them of $15 million in return for their kindness (see http://www2.bostonherald.com/news/national/ap_fraud08052002.htm). MCU subsequently installed a disaster-tolerant system configuration using StorageWorks Data Replication Manager to prevent such problems in the future (see http://www.totaltec.com/case_mcu.htm).

Select Technologies to Meet Business Needs

Once you have determined the actual needs of the business and quantified them in terms of RPO and RTO, you can choose suitable technologies to meet those business needs.

Here are some examples of recovery point objective values and technologies and techniques that can be used to meet them:

- RPO of 24 hours: Make backups at midnight every night and take them immediately off-site; recover data from the last set of backup tapes.

- RPO of 1 hour: Ship a database backup off site periodically, and ship database logs hourly to a remote site over a network; recover database to the point of the last log shipment.
- RPO of zero: Mirror data in a strictly-synchronous fashion to disks at a remote site

Here are some examples of recovery time objective values and technologies and techniques that can be used to meet them:

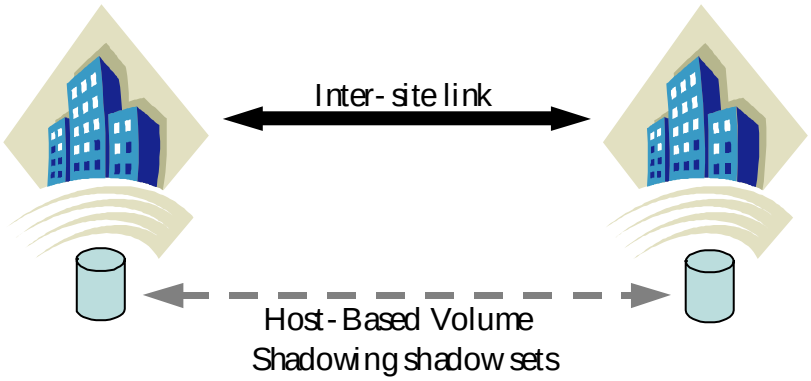
- RTO of 72 hours: Restore tapes to configure-to-order systems at a vendor DR site
- RTO of 12 hours: Restore tapes to a system at a hot site, with systems already in place there
- RTO of 4 hours: Use data vaulting to keep a copy of data at a hot site where systems are already in place
- RTO of 1 hour: Use a disaster-tolerant cluster with controller-based cross-site disk mirroring
- RTO of 10 seconds: Use:
  - A disaster-tolerant cluster with two separate, independent inter-site LAN links (thus avoiding any impact of spanning tree reconfiguration times),
  - Host-based volume shadowing to replicate the data between sites without the need for time-consuming failover operations, and
  - Clustering software with a distributed lock manager and cluster-wide file system, allowing applications to run at both sites simultaneously, rather than restarting applications at the recovery site after a failure occurs

Foundation for a Disaster-Tolerant Cluster

The goal of a disaster-tolerant cluster configuration is to survive the loss of up to one entire datacenter. One must always keep this fundamental goal in mind when designing and operating a disaster-tolerant cluster.

The crucial elements of the environment and technology that form the foundation of a disaster-tolerant cluster are:

- Two or more datacenters a “safe” distance apart
- A multi-site cluster, with cluster software for coordination among the servers
- Inter-site link for the cluster interconnect
- Data replication of some sort to keep two or more identical copies of data, one at each site



- Management and monitoring tools
  - Remote system console access or KVM system
  - Failure detection and alerting
  - Quorum recovery tool (especially for two-site clusters)
- Configuration planning and implementation assistance, and staff training
- Carefully-planned procedures for:
  - Normal operations
  - Scheduled downtime and outages
- Detailed diagnostic and recovery action plans for various failure scenarios

We'll look at these areas in more detail.

Datacenter and Site Selection

Datacenter site locations must be carefully selected to avoid hazards common to both sites, which could result in the loss of both datacenters at once.

Place datacenters a "safe" distance apart. This must be a compromise. Factors which will be involved in this decision include:

- Risks faced by each proposed datacenter site
- Performance (because inter-site latency adversely affects performance with extended inter-site distances)
- Inter-site interconnect costs
- Ease of travel between sites
- Business needs

Risk faced by a datacenter site might include:

- Fire (of a building, forest fire, gas leak, explosive materials)
- Storms (tornado, hurricane, lightning, hail, ice, snow)
- Flooding (excess rainfall, dam breakage, storm surge, broken water pipe)
- Earthquakes
- Tsunamis
- Nearby transportation of hazardous materials (by highway or rail)

- Terrorist (or disgruntled customer) with a bomb or weapon
- Enemy attack in war (are there nearby military or industrial targets?)
- Civil unrest (riots, vandalism)

A determination of what is a “safe” distance depends on the hazards one is trying to protect against:

- 1 mile: protects against most building fires, gas leak, bombs, armed intruder
- 10 miles: protects against most tornadoes, floods, hazardous material spills
- 100 miles: protects against most hurricanes, earthquakes, tsunamis, forest fires

Also, select site separation in a safe geographical location:

- Not along the same earthquake fault-line
- Not along likely storm tracks
- Not in the same floodplain, or downstream of the same dam
- Not near the same coastline (that might be hit with a hurricane or tsunami)
- Not in line with prevailing winds (that might carry hazardous materials)

Redundancy must be provided for each of:

- The datacenter itself, as well as the related facilities (air conditioning, power, user workspace, etc.)
- Data storage and external data feeds, if any
- Systems
- Network
- User access

OpenVMS Multi-Site Cluster Systems

An OpenVMS multi-site cluster system consists of multiple sites in different locations, with one or more OpenVMS systems at each site. Systems at each site are all part of the same OpenVMS cluster, and share resources under the coordination provided by the cluster software

Inter-Site Links for OpenVMS Multi-Site Cluster Configurations

An inter-site OpenVMS Cluster interconnect link must be able to handle System Communications Architecture (SCA) protocol traffic. SCA is sometimes referred to as SCS for System Communications Services, the name of the OpenVMS software that implements SCA.

Technologies that can carry SCS traffic include a bridge or a bridge/router between local area networks (LANs), or a fiber-optic link between Memory Channel II hubs. An inter-site link connecting two LANs must use bridges (or bridge-routers), because routers

alone don't pass the System Communication Services (SCS) protocol used within OpenVMS Clusters.

Minimum standards for inter-site SCS links are in the OpenVMS Cluster Software SPD (see http://www5.compaq.com/info/SPD/):

- 10 Mb minimum data rate
- Minimized packet latency
- Low SCS packet retransmit rate (less than 0.1% retransmitted). This implies:
  - Low packet-loss rate for bridges
  - Low bit-error rate for links

In addition to one or more SCS-capable cluster interconnects, one or more storage-only interconnects, such as a Fibre Channel Storage Area Network (SAN), can also link the two sites.

The inter-site links can use a variety of different technologies. Sites can be linked by:

- DS-3/T3 (E3 in Europe) or ATM circuits from a telecommunications vendor
- Microwave link: DS-3/T3 or Ethernet
- Free space optics link (short distance, low cost)
- Dark fiber where available, including:
  - ATM over SONET
  - Ethernet over fiber (10 Mb/s, Fast (100 Mb/s), Gigabit (1 Gb/s))
  - FDDI (up to 100 km)
  - Fibre Channel
  - Fiber links between Memory Channel switches (up to 3 km)
  - Wave division multiplexing (WDM), either coarse or dense wave division multiplexing (DWDM) , which can carry any type of traffic that can run over a single fiber

The bandwidth of inter-site link technologies varies, for example:

- DS-3: 45 Mb/s
- ATM: 155 or 622 Mb/s
- Ethernet: Regular (10 Mb/s), Fast Ethernet (100 Mb/s) or Gigabit Ethernet (1 Gb/second)
- Fibre Channel: 1 or 2 Gb/second
- Memory Channel: 100 MB/second
- [D]WDM: Multiples of ATM, Gigabit Ethernet, Fibre Channel, etc.

In practical terms, the minimum bandwidth required for the inter-site links is often determined by the minimum acceptable times for a volume shadowing full-copy operation to restore redundancy after a failure or scheduled outage which temporarily takes down one site or the other. This is commonly the only time the inter-site link bandwidth is fully utilized. During normal operations, the inter-site link latency is often more important than its bandwidth in its impact on system performance.

There are some important decisions to be made with regard to the inter-site link, including the following:

- What type of service will be chosen -- a circuit service provided by a telecommunications vendor, or a link that the company itself would own (for example, a private microwave link, a free space optics link, or a dark fiber link)?

- Will the bandwidth be dedicated or shared with other users? A volume shadowing full-copy operation can take significant bandwidth. Initiating a full copy operation might result in an unpleasant surprise to the other users of a shared service if sufficient bandwidth were not provided ahead of time. A full-copy operation could take an excessive amount of time to complete if sufficient bandwidth is not available.

- If vendor-provided facilities are used, will a single vendor provide all inter-site link facilities or will multiple vendors be used? Negotiating a contract for all network services can save money, but there have been recent well-documented cases where an entire vendor's network went down due to a problem with a software upgrade on their switches. Using multiple vendors can help prevent an outage due to a problem in a single vendor's network.

- Will there be multiple redundant inter-site link paths, and if so, will they be routed via diverse paths? There is a phenomenon in the industry referred to as "backhoe fade," where the signal loss happens to be exactly the same as the signal strength (and it happens when a backhoe operator accidentally severs your inter-site link). Multiple links routed on diverse paths prevents the loss of connectivity caused by such an accident.

Data Replication

Data replication between sites can be done using host-based Volume Shadowing for OpenVMS, StorageWorks Data Replication Manager (DRM), database replication, database log-shipping, or by middleware such as Reliable Transaction Router (RTR) software that can shadow transactions to two different back-end systems.

Management and Monitoring Tools

In a multi-site cluster, it is often necessary to control equipment at a remote site. One common method of system control is via serial console lines or a system console video display, keyboard, and mouse.

In the case of serial console ports, it is often useful and practical to have a log of output from the port. Such a log is readily achievable and can be extremely handy in determining the source of a failure if a piece of equipment has a problem. To be able to

access and control systems at each of the sites, disaster-tolerant cluster configurations almost invariably have some sort of system console management software and/or a keyboard video mouse (KVM) platform in place.

In addition to console access, it is crucial to have a system in place that is capable of automated failure detection and alerting. Even when redundancy is in place, it is important to be aware of and quickly repair a failed component, because the failure of a second component before the first one is repaired could cause an outage. The fact that redundancy often compensates for and masks failures makes automated failure detection and reporting even more important.

Network monitoring software needs to be in place, particularly monitoring the inter-site links. The state of all OpenVMS systems in the cluster and the state of all volume shadowing shadow sets, controller-based mirror sets, RAID sets, and so forth, need to be monitored continually.

System or network administrators should modify the LAVC$FAILURE_ANALYSIS program template provided with OpenVMS in the SYS$EXAMPLES: directory to reflect the cluster's actual LAN configuration and then use it to monitor the LAN. The LAVC$FAILURE_ANALYSIS program will generate OPCOM messages on the system consoles as it detects failures and repairs of LAN components.

Satisfactory performance is one aspect of high availability, because if the system is available but not performing well enough to do its job, it is arguably not available. So performance management is a crucial part of operating a successful disaster-tolerant cluster, and software with capabilities beyond those of the Monitor utility provided with OpenVMS is typically needed. DECamds or Availability Manager may be in place for quorum adjustment and can be very valuable for performance-related work as well.

Software should be in place to scan output from serial consoles and provide alerts if error conditions are detected. The monitoring system should be capable of notifying a system and/or network administrator by e-mail and/or pager if needed.

The following lists show possible choices of tools for use in a disaster-tolerant cluster:

- For remote system console access:
  - Heroix RoboCentral
  - CA Unicenter Console Management for OpenVMS (formerly Command/IT, formerly Polycenter Console Manager)
  - TECsys Development Inc. ConsoleWorks
  - Ki Networks Command Line Interface Manager (CLIM)

- For failure detection and alerting:
  - Heroix RoboMon and RoboER
  - CA Unicenter System Watchdog for OpenVMS (formerly Watch/IT, formerly Polycenter System Watchdog)
  - BMC Patrol

- For network monitoring:
  - HP OpenView
  - Unicenter TNG
  - Tivoli
  - ClearViSN, CiscoWorks; etc. for specific vendor products

- For performance management:
  - HP DECamds or Availability Manager
  - HP ECP (CP/Collect & CP/Analyze)
  - Perfcap PAWZ, Analyzer, & Planner
  - Unicenter Performance Management for OpenVMS (formerly Polycenter Performance Solution Data Collector and Performance Analyzer, formerly SPM and VPA)
  - Fortel SightLine/Viewpoint (formerly Datametrics)
  - BMC Patrol

The Disaster Tolerant Cluster Services (DTCS) package uses Heroix RoboCentral, RoboMon, and RoboER.

Successful monitoring and control of a disaster-tolerant cluster typically involve having at least one PC or workstation at each of the sites from which to monitor and control the cluster. If an OpenVMS system is used for monitoring, it is important that it *not* be a member of the cluster it is intended to monitor, or it will become useless if the cluster loses quorum. Note that a workstation is not required for the monitoring and control systems at each site; you can use a standalone OpenVMS server of any type and simply use a remote DECwindows display on an OpenVMS workstation (or an X-Windows display using a product such as Excursion, Hummingbird Exceed, or Reflections X on a PC) to provide a GUI interface from that server for the monitoring and control tasks.

One consideration in choosing monitoring and control tool sets is the operating system they run on, and opinions can be very strong on the choice of the operating system. Many of the products listed above can run on multiple operating systems; others can run on just one. Certain OpenVMS customers have a strong preference for monitoring and controlling OpenVMS systems only with other OpenVMS systems and refuse to use Windows or Unix platforms for monitoring and control of their OpenVMS environment. To meet the needs of these customers, HP offers a software product called CockpitMgr that is specifically designed for disaster-tolerant OpenVMS Cluster monitoring and control.

A disaster-tolerant cluster will typically need a Quorum recovery tool, especially in two-site clusters. It will be necessary to use this when quorum is lost due to failure of a site or failure of an inter-site link. With such a tool, it is possible to tell OpenVMS to re-adjust quorum and allow computing to continue.

Most sites use either the built-in graphical user interface (GUI) based quorum adjustment tools provided as a part of the Disaster Tolerant Cluster Services (DTCS) package, or the

earlier Business Recovery Server (BRS) package, or DECamds or the equivalent Availability Manager (AM) tool. DECamds and AM can be used to adjust quorum by double-clicking on a system from the Summary display, pulling down the System Fix menu, and using the Adjust Quorum option. The DTCS or BRS integrated tools use the same RMDRIVER (DECamds client) interface that is used by DECamds and Availability Manager.

While it is theoretically possible to use the IPL 12 software interrupt routine to adjust quorum (issuing a Q command at the IPC> prompt), this doesn't work reliably in multi-processor machines due to timing issues, so this is not a practical option in practice.

Configuration Planning, Implementation Assistance, and Staff Training

For configuration planning, implementation assistance, and staff training, HP recommends the Disaster Tolerant Cluster Services (DTCS) package.

DTCS includes assistance in developing procedures for normal operations, scheduled downtime and outages, and detailed diagnostic and recovery action plans for various failure scenarios, so that the staff will know exactly what troubleshooting and repair actions to take in any crisis situation.

Planning for Disaster Tolerance

Always remember that the goal of planning for disaster tolerance is to continue operating despite loss of an entire datacenter. The following conditions must exist:

- User access to both sites
- Network connections to both sites
- Operations staff at both sites

The business can't depend on *anything* that is *only* at one site.

Also, be sure to plan for continued operation *after* a disaster. It is likely that the surviving site will have to operate alone for a long period before the damaged site can be repaired or replaced. Constructing a replacement data center could easily take nine months to a year or more. So it is important to provide enough computing capacity within each site to run the business alone if either site is lost, and to continue to handle the workload as it grows, taking into consideration the possible range of workload growth rates that might occur.

Having three full datacenters is an option to seriously consider, because this leaves two redundant sites after a disaster, and leaves you with two-thirds of your computing capacity in place after a disaster instead of just one-half.

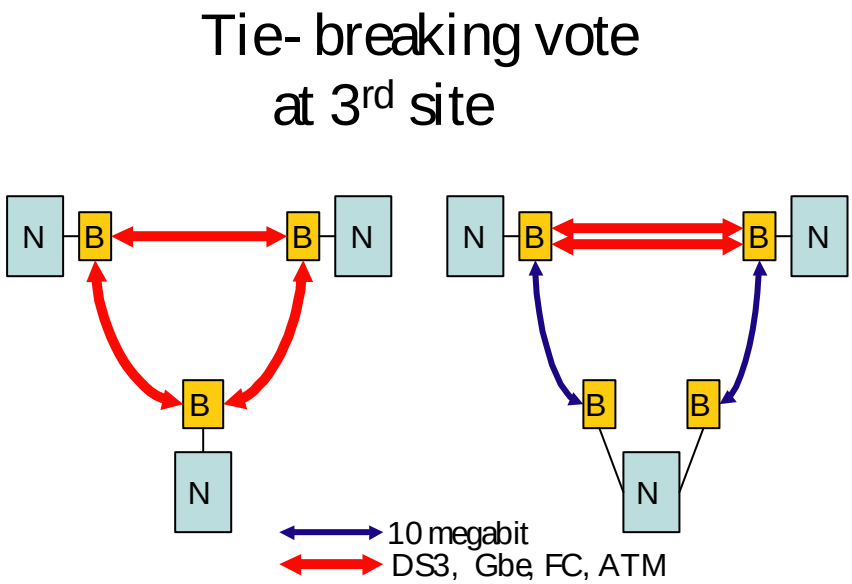
Quorum Configurations in Multi-Site Clusters

Clusters that allow shared access to data typically use a quorum scheme of some sort to arbitrate access to the shared resources. The quorum scheme is based on familiar parliamentary procedures, where a simple majority (just over half) of voting members must be present for business to be transacted.

In the event of a communications failure within the cluster, systems in the minority voluntarily suspend their processing, while systems in the majority can continue to process transactions. If there is an even number of votes, and a communications failure splits the cluster in half, neither half can continue processing, because neither half can achieve quorum.

There are a number of options for assigning system votes and quorum in a multi-site cluster, such as:

- Three sites, with equal votes in two sites. This is intuitively ideal, and is the easiest to manage and operate. The third site contains a quorum node, which provides a tie-breaking vote. Unfortunately, this is often hard to do in practice, due to the cost of inter-site links beyond on-campus distances. To mitigate the inter-site link costs, one could use links to the third site as the backup for the main inter-site link. This works as long as the links to the third site are high-bandwidth links bridged together at the third site. Alternatively, to lower costs, one could use two less-expensive, lower-bandwidth links (perhaps at the minimum supported bandwidth of 10 megabits) to the third site.



- Two sites. This is the most common and most problematic. How do you arrange votes? Balanced? Unbalanced? If votes are balanced, you will need to be prepared to take manual action to recover from the loss of quorum which will result when either site or the inter-site link fails.

To create a two-site cluster with unbalanced votes, one simply gives more votes to systems at one site. The site with the most votes can continue to operate without human intervention in the event of loss of the other site or loss of the inter-site link. The site with fewer votes pauses on a failure and requires manual action to continue after loss of the other site. Unbalanced vote configurations are very common in remote-shadowing-only clusters (where the data is shadowed to a remote site, but the configuration is not fully disaster-tolerant), and 0 votes is a common choice for the remote site in this case.

A common mistake in practice with multi-site clusters with unbalanced votes is to give more votes to a primary site, and leave the standby site unmanned. The result is that the cluster can't run without the primary site, unless there is human intervention, which is unavailable at the unmanned standby site.

Another side effect of unbalanced votes is that the decision of which site will continue in the event of an inter-site link failure is pre-determined by the selection of vote counts, which can leave one subject to the rare possibility of data loss due to a "creeping doom" scenario, which we will discuss later.

To create a two-site cluster with balanced votes, one gives an equal number of votes to systems at each site. This type of configuration will require manual action to resolve the quorum loss that occurs when one site is lost or the inter-site link fails. This quorum adjustment can be done either with DECamds or Availability Manager, or with the tools provided with the BRS or DTCS packages that use the RMDRIVER interface provided with DECamds.

Note that while operating the cluster, using the REMOVE_NODE option to SHUTDOWN.COM (for OpenVMS Version 6.2 or later) or, equivalently, using the SYSMAN command SHUTDOWN NODE with the /REMOVE_NODE qualifier when taking down a node effectively "unbalances" the configuration of votes. For this reason, it should be used with special care in a disaster-tolerant cluster configuration.

Optimal Sub-Cluster Selection

The systems in an OpenVMS cluster must all be able to communicate with one another. When a failure occurs that disrupts connectivity between nodes that are members of the cluster, OpenVMS Cluster software must determine which subset of nodes will remain in the cluster.

The Connection Manager within OpenVMS compares potential node subsets that could make up the surviving portion of the cluster, using the following criteria:

- Pick the subset with the most votes.
- If the number of votes is tied, pick the subset with the most nodes.
- If the number of nodes is also tied, OpenVMS "arbitrarily" picks a winning subset based on a comparison of SCSSYSTEMID values among the set of nodes with the most-recent cluster software revision.

Volume Shadowing for OpenVMS Multi-Site Considerations

Most OpenVMS disaster-tolerant clusters use the host-based Volume Shadowing for OpenVMS software to perform data mirroring between sites.

Shadow Copy Algorithm

The full-copy algorithm for volume shadowing may seem non-intuitive. The SHADOW_SERVER process does the copy I/O operations. Starting at the beginning of the disk, and going to the end, in segments of 127 blocks, it will:

1. Read from the source disk.
2. Compare the data read with the data on the target disk.
3. If the data is different, write to target disk, then go to Step 1.

Volume shadowing uses no double-buffering or other speed-up tricks in this operation. This somewhat odd algorithm ensures correct results even in cases such as on a system disk when a node is booting or writing a crash dump while a shadow copy operation is taking place.

One can speed shadow copy operations in a disaster-tolerant cluster if one understands the implications of this algorithm. One implication is that a shadow copy completes fastest if the data is identical beforehand. Fortunately, this is the most-common case – returning a former member to a shadow set.

If the OpenVMS MSCP server is being used to allow access to the disks at a remote site (rather than accessing the disks directly by means of an inter-site Fibre Channel link), one should be aware that remote MSCP-served write operations take a minimum of two round trips between sites. However, thanks to an optimization in the LAN implementation of the SCS protocol, read operations can be done in a minimum of one round trip. Therefore, it is more efficient to have the SHADOW_SERVER doing the shadow copy running on a node at the target site, so the remote operations it must perform are read operations, rather than write operations. One can temporarily change the value of the dynamic SYSGEN parameter SHADOW_MAX_COPY to zero on nodes where a shadow copy thread is not desired, initiate the shadow copy, then change the values of SHADOW_MAX _COPY back to their previous settings.

Because volume shadowing copies each disk sequentially from beginning to end, in 127-block segments, and because a given shadow-copy operation is typically limited by the latency of the 127-block disk I/Os at each end, a full-copy operation between sites can

complete faster in practice if you have many shadow sets copying in parallel, so as to have a greater chance of fully-utilizing the bandwidth of the (typically costly) inter-site link. It is often necessary to raise SHADOW_MAX_COPY from its default value of 4 to allow enough shadow-copy threads to shadow-copy all of the shadow sets in parallel.

A shadow full-copy operation can often complete faster if there are more, smaller disk units to copy rather than a smaller number of larger units. This means that one should not necessarily use the largest-capacity disk models available in a disaster-tolerant cluster. And rather than combining disks into controller-based stripesets or RAID sets, full-copy elapsed times are generally shorter if you shadow individual disk units with volume shadowing. If necessary, you can then combine them into larger units using host-based RAID software to form host-based RAID 0+1 arrays. In fact, tests have shown that an individual disk, when divided at the controller level into four, equal-sized partitions and presented to OpenVMS to use as four individual disks, and when these four mini-disks are copied in parallel, the elapsed time for a full-copy operation is cut to only 40% of the time to shadow-copy the disk as a whole.

Protecting Shadowed Data

Shadowing keeps a generation number in the storage control block (SCB) on shadow set member disks. The SCB is stored in virtual block number (VBN) 1 of the BITMAP.SYS file.

When various shadow set events occur, such as mounting or a membership change, volume shadowing increases the generation number. The generation number is designed to monotonically increase over time, and never decrease. The actual implementation is based on the 8-byte OpenVMS timestamp value. The generation number will be increased to one of the following values:

- Current time value
- Current generation number plus 1. If the generation number already represents a time in the future for some reason (such as a time skew among cluster member clocks), then it is simply incremented

The new value is stored on all shadow set members when the increase occurs.
As a result, the generation number in the SCB on removed members will tend to gradually fall farther and farther behind that of current members.

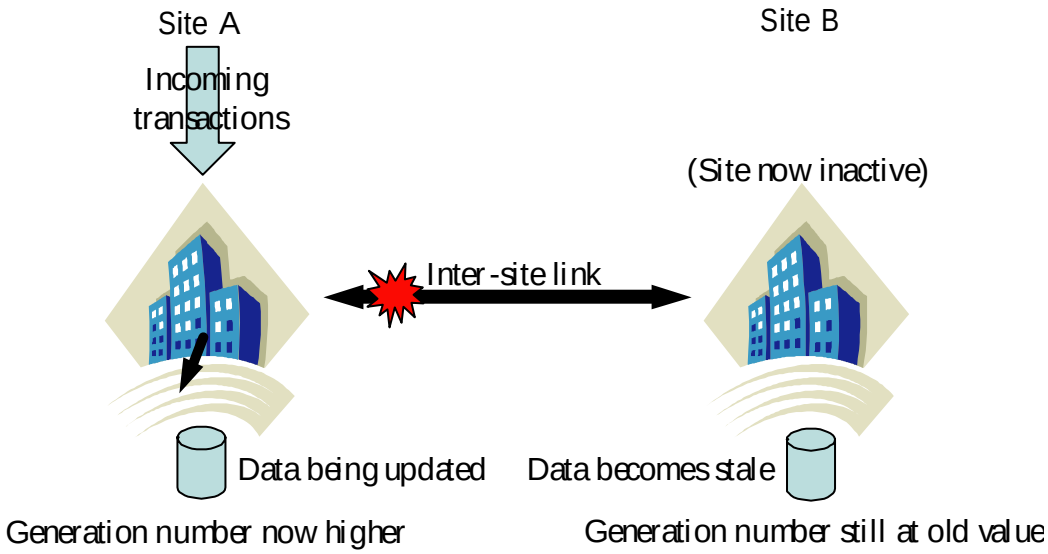
In comparing two disks under normal circumstances, a later generation number should always be on the more up-to-date member.

“Wrong-Way Shadow Copy” Scenario

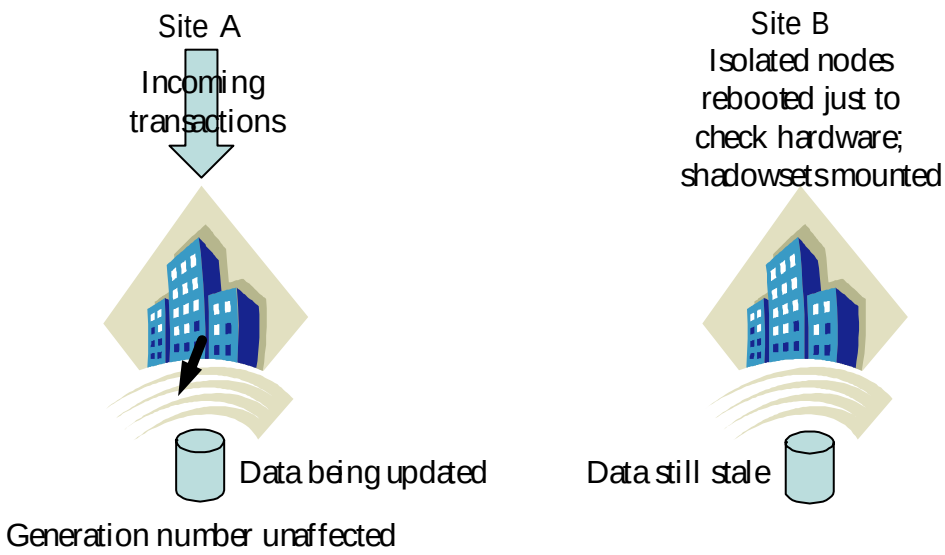
A nightmare scenario for a disaster-tolerant cluster would be a shadow copy that goes in the wrong direction, copying old data over the up-to-date copy of the data. The way volume shadowing software keeps and uses generation numbers normally prevents this

from happening, but, lacking an understanding of how volume shadowing uses generation numbers, one might mistakenly create circumstances under which a “wrong-way copy” could occur.

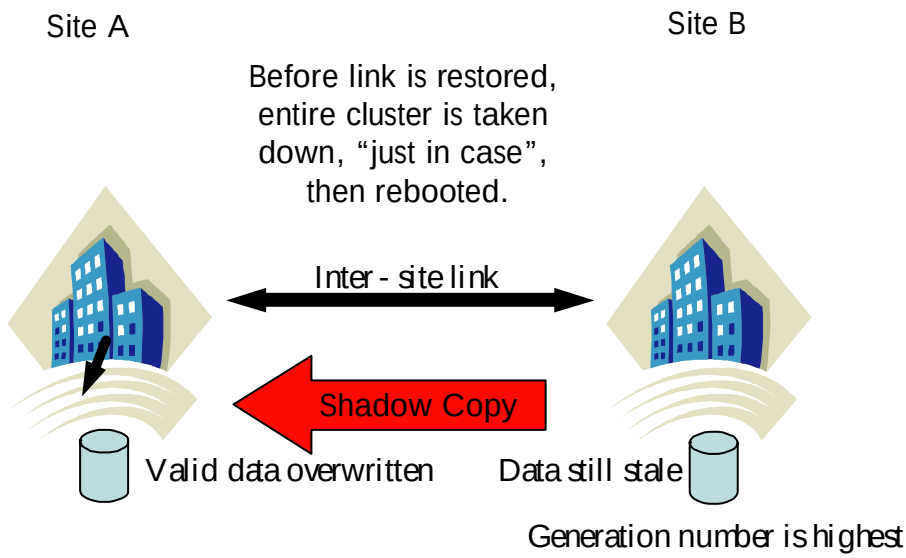
In a real-life example recounted by one disaster-tolerant cluster site, an inter-site link failure occurred. Due to an unbalanced-votes configuration favoring it, Site A automatically continued to run. Volume shadowing increased the generation numbers on the Site A disks after the removing Site B disks from the shadow sets.



Site B was brought up briefly by itself, reportedly to verify the operation of its hardware. Because the inter-site link was still down, volume shadowing at that site could not see the Site A disks, so volume shadowing mounted the shadow sets within Site B with Site B disks only. At the time of the Mount operation, shadowing bumped the generation numbers on the Site B disks. Because the system time is used for the new value, the generation numbers on the Site B disks were now greater than the generation numbers on the Site A disks.



The inter-site link was repaired. For the sake of caution, both sites were taken down and rebooted at once. Volume shadowing compared the generation numbers on the disks, concluded that the disks at Site B were more current, and copied them over Site A's disks, resulting in data loss.

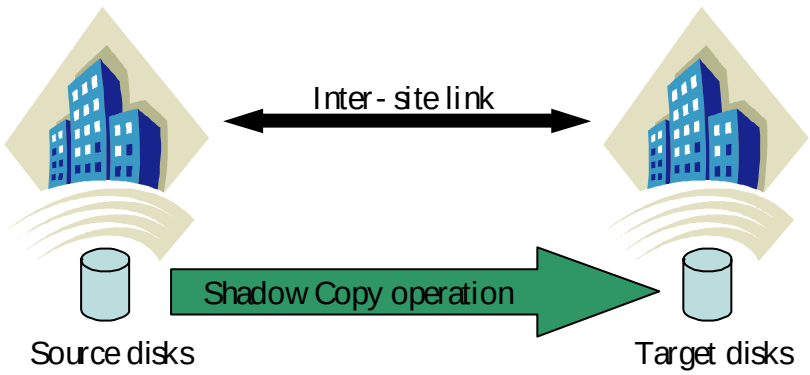


If shadowing can't "see" a later disk's SCB (that is, because the site or link to the site is down), it may use an older member and then update the generation number to a current timestamp value. To help prevent this problem, one can use the /POLICY=REQUIRE_MEMBERS qualifier on $MOUNT commands for cross-site shadow sets to prevent a mount from succeeding unless all of the listed members are present for volume shadowing to compare their generation numbers.

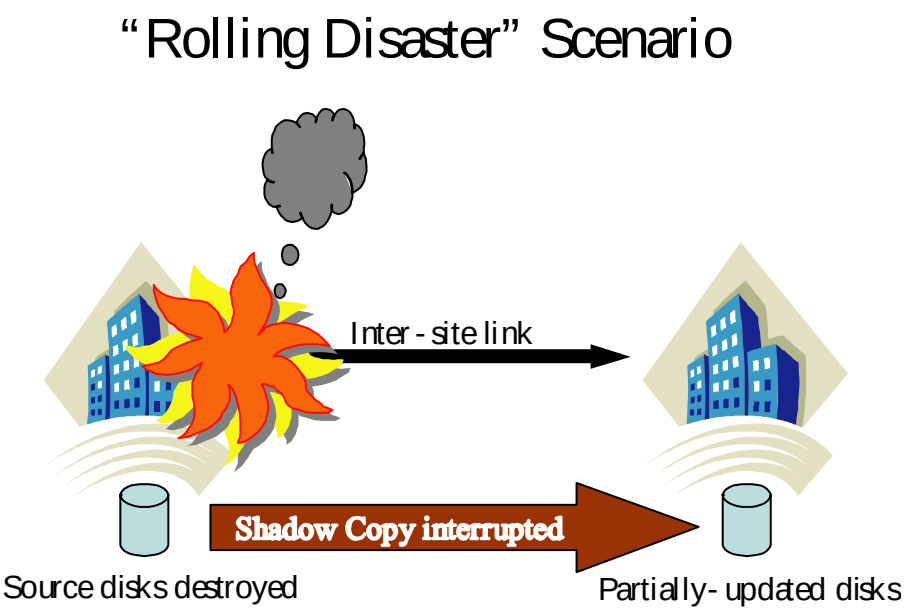
One can also use the /POLICY=VERIFY_LABEL on $MOUNT commands to help prevented unwanted shadow copy operations. This qualifier means that the volume label on a disk must be SCRATCH or it won't be added to the shadow set as a full-copy target. This implies that a system manager must manually issue an $INITIALIZE command with the label SCRATCH on a disk before the disk t can be re-added to a shadow set.

"Rolling Disaster" Scenario

This is another rare but theoretically possible case where data might be lost, so care should be taken to prevent it from happening. In this scenario, a disaster or outage occurs that causes one site's data to become out-of-date.



Perhaps there is a scheduled power shutdown at one site to safely work on the power distribution unit. Then, the site that had the power shutdown has regained power and a full-copy operation, to update a shadow set, is in progress.
While performing a full-copy operation, a disaster takes out the primary site.



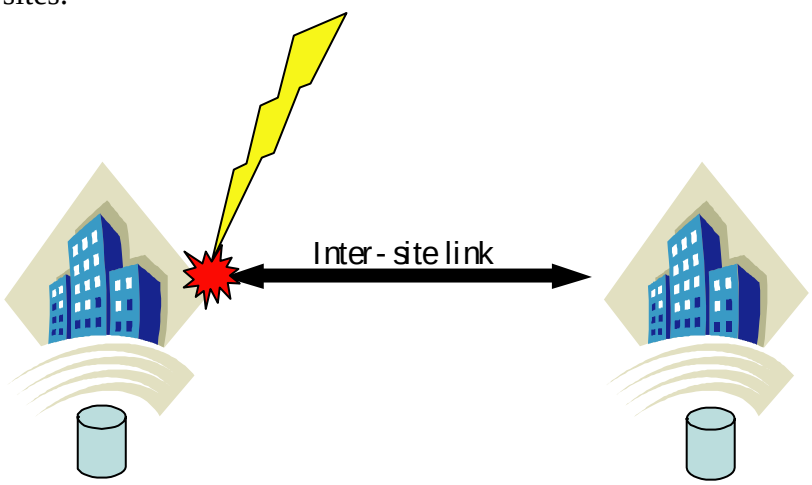
Techniques for avoiding data loss due to rolling disaster include:

- Keep a copy (backup, snapshot, or clone) of the out-of-date copy at the target site instead of over-writing the only copy there. You could perform the full-copy operation to a different set of disks from the originals. In either case, the surviving copy will be out-of-date, but at least you'll have *some* copy of the data.
- Keep a third copy of the data at a third site by using three-member shadow sets. This is the only way to ensure there is no data lost in a rolling disaster scenario.

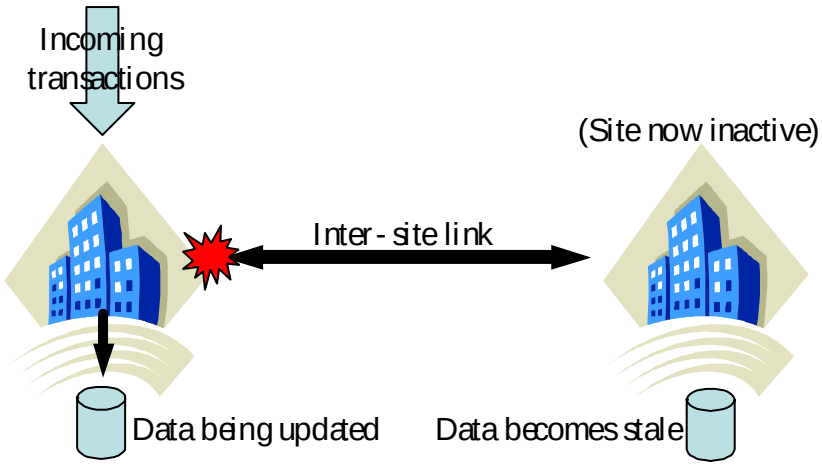
"Creeping Doom" Scenario

One more unlikely but theoretically-possible scenario is known as creeping doom. In this scenario, a failure starts as a small, hidden failure at one site, but then grows to engulf and destroy the entire site.

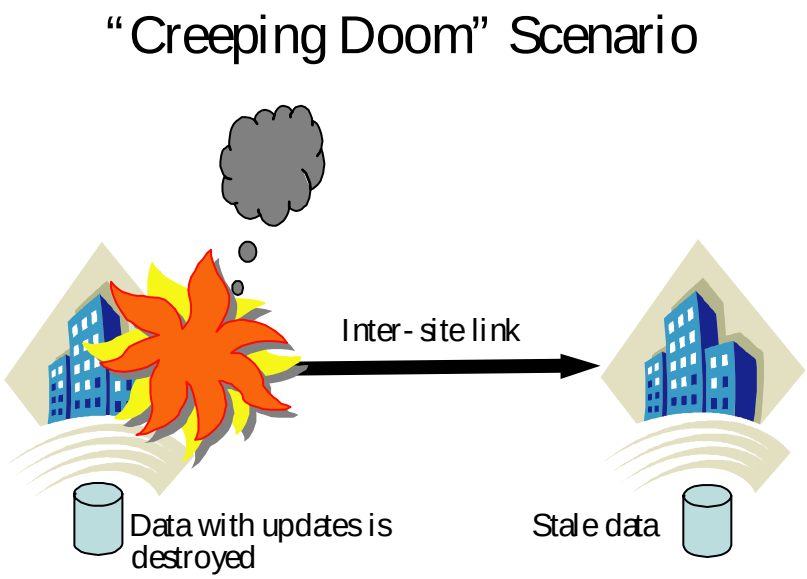
In this scenario, the first symptom is a failure of the link or links between two sites.



In our example, a lightning strike hits the network room, taking out all the inter-site links. The inter-site link failure forces a choice of one of the two datacenters to continue processing transactions. Transactions then continue to be processed at the chosen datacenter, updating the data on the disks at that site. In this case, for the sake of illustration, we'll assume we chose the left-hand site (the one where lightning struck) to continue processing transactions.



In the "creeping doom" scenario, the same failure which initially caused the inter-site links to go down expands to destroy the entire datacenter. In our example, the fire that lightning started in the network room expands to engulf the entire datacenter.



Transactions processed after the choice of the "wrong" datacenter are thus lost. Commitments implied to customers by those transactions are also lost.

Techniques for avoiding data loss due to "creeping doom" include:

- Using a tie-breaking vote at a third site. This helps in many (but not all) cases.
- Having humans (who can smell smoke, for example) involved in the choice of which datacenter will continue processing transactions after a disaster, instead of

making this choice beforehand (for example, with an imbalance of votes between sites)
- Having a third copy of data at a third site. This is the only way to ensure that there is no data lost in a “Creeping Doom” scenario
-

System Management of a Disaster-Tolerant Cluster

Certain techniques can make system management of a disaster-tolerant cluster easier.

Managing Multiple System Disks

Most disaster-tolerant clusters will contain multiple system disks, so minimizing the system management workload (and risk of errors) associated with having multiple system disks is important.

The system manager should create a “cluster-common” disk volume which is a cross-site shadow set. This shadow set should be mounted during system startup in SYLOGICALS.COM. Put all cluster-common files onto that cluster-common disk, and define logical names in SYLOGICALS.COM to point to their location on the cluster-common disk. Files that are good candidates for inclusion on the cluster-common disk include SYSUAF, RIGHTSLIST, the Queue file, the LMF database, etc.

I recommend putting system startup files on the cluster-common disk, also, and replacing startup files on all system disks with a command procedure that calls the “real” command procedure on the cluster-common disk. For example, SYS$STARTUP:STARTUP_VMS.COM might contain only the single line:

```
$ @CLUSTER_COMMON:SYSTARTUP_VMS
```

where the logical name “CLUSTER_COMMON” was defined in SYLOGICALS.COM to point to a directory on the cluster-common disk.

To allow for different startup behavior on different nodes in a common startup file, include tests for the node name in the common startup file, as needed. For example, one might use the following DCL code to handle a special case for a node named GEORGE:

```
$ NODE = F$GETSYI(“NODENAME”)
$ IF NODE .EQS. “GEORGE” THEN ...
```

It also helps reduce work if you create a MODPARAMS_COMMON.DAT file on the cluster-common disk, which contains system parameter settings common to all nodes. For multi-site or disaster-tolerant clusters, create one of these for each site. Include an AGEN$INCLUDE_PARAMS line in each node-specific MODPARAMS.DAT to include the common parameter settings. With this technique, most MODPARAMS.DAT files for individual nodes will contain only a few lines, defining a very few parameters such as

SCSSYSTEMID and SCSNODE. Then, most parameter changes can be made by editing the common files instead, then running AUTOGEN on each node.

You can use a system disk cloning technique to replicate system disks and avoid doing "n" upgrades for "n" system disks. In this cloning technique, you first create a "master" system disk with system roots for every node in the cluster. Then, you use the Backup utility to create clone system disks from this master system disk. Subsets of the nodes are booted from each clone system disk. To minimize disk space requirements, move the dump files off the system disk for all nodes.

Before an upgrade, save any important system-specific information from clone system disks into the corresponding roots on the master system disk. Basically, this will include anything that is in SYS$SPECIFIC:[*]. Examples would include: ALPHAVMSSYS.PAR, MODPARAMS.DAT, AGEN$FEEDBACK.DAT, ERRLOG.SYS, OPERATOR.LOG, ACCOUNTNG.DAT. Perform the upgrade on the master system disk, and then use Backup to copy the master to the clone disks again.

Hardware Configuration at Each Site

Most disaster-tolerant clusters will include provisions for identical storage at each site so that all the data can be shadowed. Often, disaster-tolerant clusters will have identical server configurations, as well. It can help diagnostic and troubleshooting efforts greatly if:

- The systems, storage, and network configuration at each site is identical in both physical and logical layout. Having the systems and storage at each site configured identically also helps provide predictable performance of either site when running alone.
- Node names, controller names (and CLI prompts), and device unit numbers are chosen so as to readily indicate the site where the equipment is located. In addition, it can be helpful if device unit numbers also indicate the controller pair to which they are attached to (or at least the cabinet where they're located).

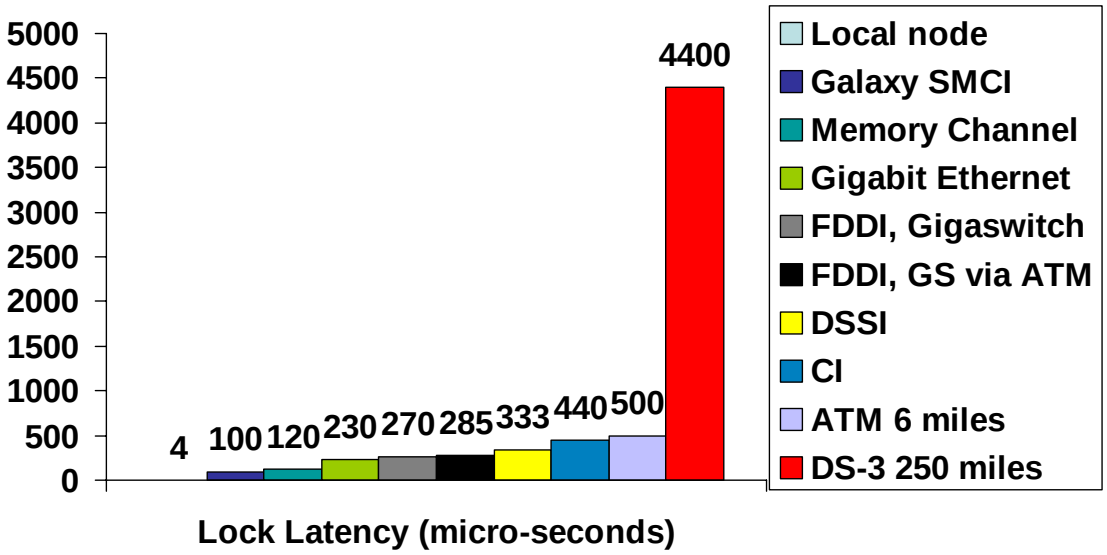
Long-Distance Disaster-Tolerant Clusters

The OpenVMS SPD specifies that OpenVMS Cluster Software supports distances of up to 150 miles (250 km) between sites, or up to 500 miles (833 km) with the DTCS or BRS packages. The limits are based on inter-site latency.

Latency due to the speed of light becomes significant at higher distances. Some rules of thumb which can be used to predict the approximate latency due to the speed of light are:

- About 1 ms per 100 miles, one-way or
- About 1 ms per 50 miles, round-trip latency

The actual circuit path length can be longer than the highway mileage between sites. For example, one disaster-tolerant configuration with a 130-mile highway mileage distance between sites had a 252-mile actual circuit path length.



Inter-site latency can affect the performance of I/O and locking operations that span the sites. I/Os that cross the inter-site link include write I/Os to remote shadow set members. Lock operations that may cross the inter-site link include lock requests ($ENQ and $DEQ), directory lookups, $GETLKI requests, and deadlock searches.

It is important to differentiate between latency and bandwidth. You cannot avoid the speed-of-light latency effects over long distances. Buying a higher-bandwidth link (often referred to loosely as a higher-speed link) doesn't mean lower latency.

There are three main ways in which applications are typically run in a disaster-tolerant cluster:

1. Hot primary/cold standby: All applications normally run at the primary site. The second site is idle, except for volume shadowing, until the primary site fails, then it takes over all processing.
   - Performance will be good (all-local locking)
   - Fail-over time will be poor, and risk high (standby systems are not active and thus not being tested until the failover needs to occur)
   - This wastes computing capacity at the remote site

2. Hot/hot but with different workloads: All applications normally run at one site or the other, but not both; data is shadowed between sites, and the opposite site takes over upon a failure.
   - Performance will be good (all-local locking)
   - Fail-over time will be poor, and risk moderate (standby systems are in use, but the specific applications are not active and thus not being tested from that site)

- Second site's computing capacity is actively used

3. Uniform workload across sites: All applications normally run at both sites simultaneously. The surviving site takes over all load upon a failure. This is the scheme that "feels right" if one wants to take full advantage of the load-balancing abilities of an OpenVMS Cluster configuration, but:
   - Performance may be affected (because there will be some remote locking), especially if the inter-site distance is large
   - Fail-over time will be excellent, and risk low (standby systems are already in use running the same applications, thus constantly being tested)
   - Both sites' computing capacity is actively used

Real-Life Examples of Disaster-Tolerant Clusters

Credit Lyonnais fire in Paris, May 1996

Credit Lyonnais had a remote-shadowing cluster set up when their building suffered a fire. Shadowing the data to the remote site using Volume Shadowing for OpenVMS saved their data. The fire occurred over a weekend, and the existence of the DR site coupled with quick procurement of replacement hardware allowed the bank to reopen on Monday.

Online Stock Brokerage

At 2 a.m. on Dec. 29, 1999, an active stock market trading day, the audio alert on a UPS system alarmed a security guard on his first day on the job. He pressed the emergency power-off switch, taking down the entire datacenter. Because a disaster-tolerant OpenVMS cluster was in place, the cluster continued to run at the opposite site, with no disruption. The brokerage ran through that stock-trading day on the one site alone, and performed a shadow full-copy operation to restore redundancy in the evening, after trading hours. They procured a replacement for the failed security guard by the next day.

Commerzbank on 9/11

Commerzbank had a datacenter near the World Trade Center towers. Generators took over after the power failure caused by the terrorist attacks, but dust & debris eventually caused the air conditioning units to fail. A disaster-tolerant cluster was in place, and the data was shadowed to disks at a remote site 30 miles away. One OpenVMS server continued to run despite 104° temperatures, running off of the copy of the data at the opposite site after all the local disk drives had succumbed to the heat.
See http://www.openvms.compaq.com/openvms/brochures/commerzbank/.

eSpeed, 9/11

The company arguably hardest-hit in the 9/11 terrorist attacks was the Cantor-Fitzgerald family of companies. For more on the story of this company's tragic losses and their courageous recovery, see http://www.espeed.com/ and follow the links under "Our Story" and "View Our Stories".

Summary

This article has described the concepts of disaster tolerance and disaster recovery and the components available for designing, configuring, and managing a disaster tolerant OpenVMS Cluster system. Recommendations for setting quorum for a multi-site cluster system were given, and in-depth explanations of the volume shadowing algorithm and its implications for a disaster-tolerant configuration were presented. Many management tools were presented, and the HP product and service offering, Disaster Tolerant Cluster Services (DTCS), was briefly described. While disaster-tolerant configurations share many similarities, by their very nature, they are complex configurations that require a high level of expertise to plan, configure, and maintain. HP offers the software, services, and tools to create and maintain these configurations.

Keith Parris

E-mail: Keith.Parris@hp.com
Web: http://encompasserve.org/~parris/ and http://www.geocities.com/keithparris/