Microsoft® Windows Server System™

Server Clusters : Geographically Dispersed Clusters

For Windows 2000 and Windows Server 2003

Microsoft Corporation

Published: November 2004

Abstract

This document describes what a geographically dispersed server cluster is and how Microsoft Windows server cluster can be used to provide a disaster-tolerant environment for mission critical data and applications.

Microsoft

Contents

Introduction **1**

Multi-site Configurations **2**

Multi-site NLB Configurations 2

Multi-site MSCS Configurations 3

What Is a Geographically Dispersed Cluster? 4

Data Replication 5

Application Failover 6

Deploying Geographically Dispersed Clusters **8**

Qualified Configurations 8

MSCS Technology Restrictions 8

Other Considerations 9

Basic Architecture 10

Operational and Deployment Procedures 10

Server Cluster Parameters 11

Applications on Multi-site Clusters 11

Server Cluster Features in Windows Server 2003 ..... **13**

Three-Site Majority Node Set Quorum in Geographically Dispersed Clusters to Facilitate Automatic Failover **15**

Summary **17**

Related Links **18**

Introduction

More and more enterprises are running mission-critical applications that are fundamental to the core business. Failure of those applications can be potentially disastrous to the business.

As Windows server operating systems become increasingly accepted in the large scale and high-end mission-critical parts of organizations, the requirements for disaster tolerance and business continuance become more and more important. The goal of building complete solutions in this environment is to ensure that there is *no single point of failure*. In other words, the loss of a single component cannot cause an application or service to become unavailable. There are many components involved in providing this level of availability, and they can be categorized as:

- *Storage failures*: There are many ways to protect against failures of individual storage devices using techniques such as Redundant Array of Independent Disks (RAID). Storage vendors (such as Compaq, EMC, and Hitachi) provide hardware solutions that support many different types of hardware redundancy for storage devices, allowing devices, as well as components in the storage controller itself, to be swapped out as necessary without losing access to the critical asset, the data itself. Software solutions (such as those from Veritas) also exist that provide similar capabilities running on the Microsoft Windows platform.
- *Network failures*: There are many components to a computer network and there are many topologies that provide highly available connectivity. All types of networks need to be considered including the client access networks, the management networks, as well as the storage fabrics (storage area networks) that link the computers to the storage units.
- *Computer failures*: Many enterprise level server platforms (such as those from Compaq, Dell, Unisys, and HP) provide redundancy inside the computer itself, such as redundant power supplies and fans. More and more vendors allow such things as PCI cards and memory to be swapped in and out without taking the machine out of service. There are also classes of machines (such as those produced by Stratus) that provide fault tolerance by replicating the entire machine and linking multiple CPUs and memory in lock-step. There are, however, occasions where, however much redundancy a machine has, the complete machine will fail or the machine has to be taken down for routine maintenance such as upgrading the software. In these cases, cluster technologies such as Microsoft Cluster Server (MSCS) and Network Load Balancing (NLB) provide redundancy to enable applications or services to continue automatically either through failover of the application or by having multiple instances of the same application available for client requests.
- *Site Failures*: In the most extreme cases, a complete site can fail, either due to a total loss of power, through a natural disaster, or due to other issues. More and more businesses (especially those in areas where natural disasters are a threat) recognize the value of deploying mission critical solutions across multiple geographically dispersed sites.

Multi-site Configurations

There are many different reasons for deploying nodes across multiple data centers in different geographic locations. The two major reasons are:

1) To provide “local access” to clients spread across a wide geographic area. For example, a Web portal may have data centers strategically placed across the globe to provide localized access to a set of services for consumers. For instance, in the United States, a portal may have data centers on the west coast and east coast.

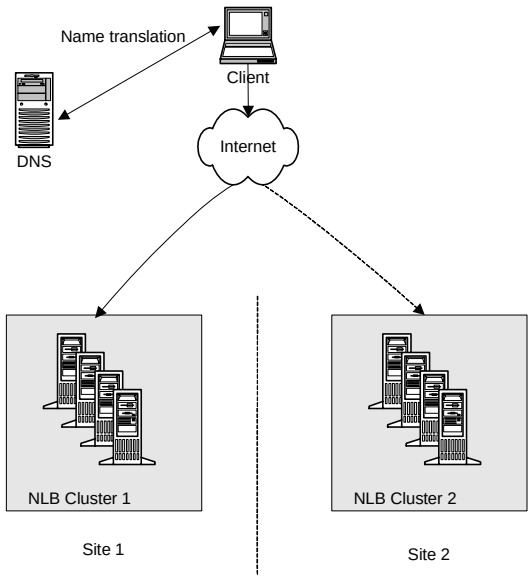
2) To provide a disaster tolerant solution so that in the event of a single site loss or failure, the mission critical application or services can continue.

Clusters are defined as a set of nodes that together, provide a highly available and highly scalable platform for hosting applications. Windows provides two different cluster technologies aimed at different application styles. *Server clusters* provides a highly available environment for long running applications and services such as Microsoft SQL Server™ by providing failover support. If a node fails that is hosting an application, the application is restarted or failed over to another node. NLB provides a highly scalable and highly available platform for applications that are scaled out by running multiple copies of the same application against the same data set, providing load balancing of client requests across the set of nodes.

Multi-site NLB Configurations

A single NLB cluster is not typically deployed across multiple sites (for a number of reasons the technology is best suited to a broadcast-capable LAN environment). Instead, multiple, independent NLB clusters are typically deployed, one at each site. The sites are then “federated” into a single service to clients using features such as DNS round-robin to “load balance” client requests across the different sites. Internally, NLB can be used to load balance requests sent to a given site amongst the nodes in that site.

Figure 1. A multi-site NLB configuration



In the example in Figure 1, the client machine uses DNS to translate the service name to an IP address. Depending on the IP address that the client received, the request is sent to either site 1 or site 2.

More complex techniques can be employed to provide *global load balancing* to direct clients to their closest site to optimize response times. These techniques are beyond the scope of this document.

Regardless of how the requests are load balanced across the sites, in this type of configuration, each site is providing the same set of services (for example, the same web site). The sites are essentially clones of each other. Although they do not necessarily have identical hardware and configurations at each site, clients will receive the same response to a given query, regardless of the site that they are directed to. These types of solutions provide the potential for optimizing client access, as well as providing disaster tolerance; if one site fails, the other site can respond to client requests (albeit with a potentially longer response time for individual requests).

Multi-site MSCS Configurations

MSCS clusters host applications that use failover to achieve high availability, for example SQL Server database instances can be made highly available using failover. The failover mechanism is automatic; MSCS provides health monitoring capabilities and if MSCS detects that an application is not responsive, the application has failed, or the node hosting the application has failed, MSCS takes care of ensuring that the application fails over to another node in the cluster. MSCS provides very strong guarantees that, regardless of failures of nodes or communication links, only a single instance of a given application is running at any point in time. This is very important to avoid data corruption or inconsistent results to the clients. To provide the strong guarantees, all the nodes that can host the application must be in the same MSCS cluster. Therefore, to provide disaster tolerance for these kinds of application, a single MSCS cluster must be “stretched” across multiple sites.

The goal of multi-site MSCS configurations is to ensure that loss of one site in the solution does not cause a loss of the complete application. These configurations are not designed to provide “local” access to client requests. For business continuance and disaster tolerant configurations, sites are typically up to a few hundred miles apart so that they have completely different power, different communications infrastructure providers and are placed so that natural disasters (e.g. earthquakes) are extremely unlikely to take out more than one site.

Fundamentally, a multi-site MSCS configuration has to solve two specific issues:

- How to make sure that multiple sites have independent copies of the same data.
- Each site must have its own copy of the data since the goal of the solution is that if one site is lost, the applications can continue. Where the data is read-only, this does not present much of a problem, the data can be copied and an instance of that data can be hosted at each site. The issue comes when the data itself is changing; how do changes made to the data at one site make their way to the other sites so that in the event that the first site fails, the changes are available?. In other words how are changes replicated across sites?

In the event of the failure of one site, how does the application get restarted at another site? Once the data is replicated at multiple sites the issue becomes: how is the application started on an alternate site if the site that was running the application fails?

What Is a Geographically Dispersed Cluster?

A geographically dispersed cluster is an MSCS cluster that has the following attributes:

- Has multiple storage arrays, at least one deployed at each site. This ensures that in the event of failure of any one site, the other site(s) will have local copies of the data that they can use to continue to provide the services and applications.
- Nodes are connected to storage in such a way that in the event of a failure of a site or the communication links between sites, the nodes on a given site can access the storage on that site. In other words, in a two-site configuration, the nodes in site A are connected to the storage in site A directly, and the nodes in site B are connected to the storage in site B directly. The nodes in site A can continue without accessing the storage on site B and vice-versa.
- The storage fabric or host-based software provides a way to mirror or replicate data between the sites so that each site has a copy of the data. (Different levels of consistency are available, see the *Data Replication* section below).

Figure 2 shows a simple two-site cluster configuration.

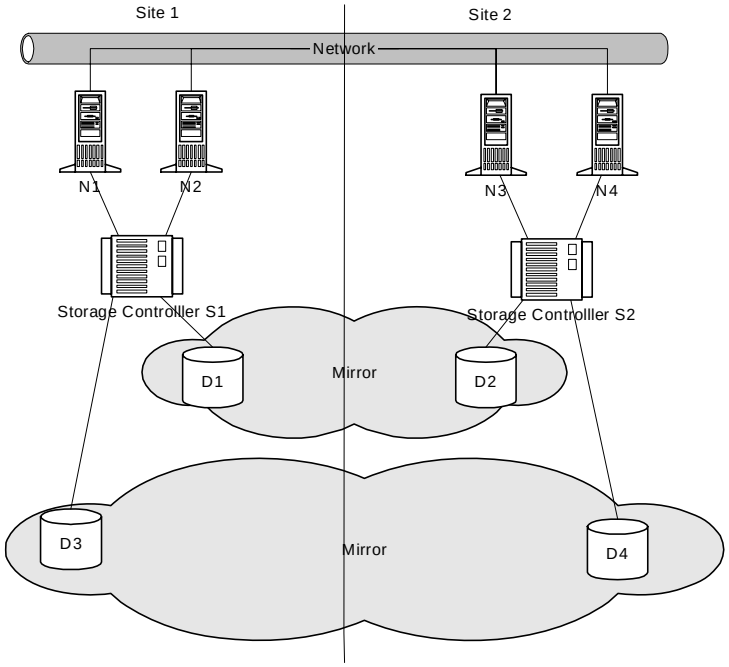


Figure 2. A simple two-site configuration

In this example, nodes N1 and N2 are connected to one array (S1), N3 and N4 are connected to another array (S2). The storage arrays conspire to present a single view of the disks spanning both arrays. In other words, disks D1 and D2 above are combined into a single logical device (via mirroring either at the controller level or the host level), to present what looks like a single device that can failover between N1, N2, N3 and N4.

There are other disaster recovery solutions that are NOT classified as geographically dispersed clusters, as shown in Figure 3 below.

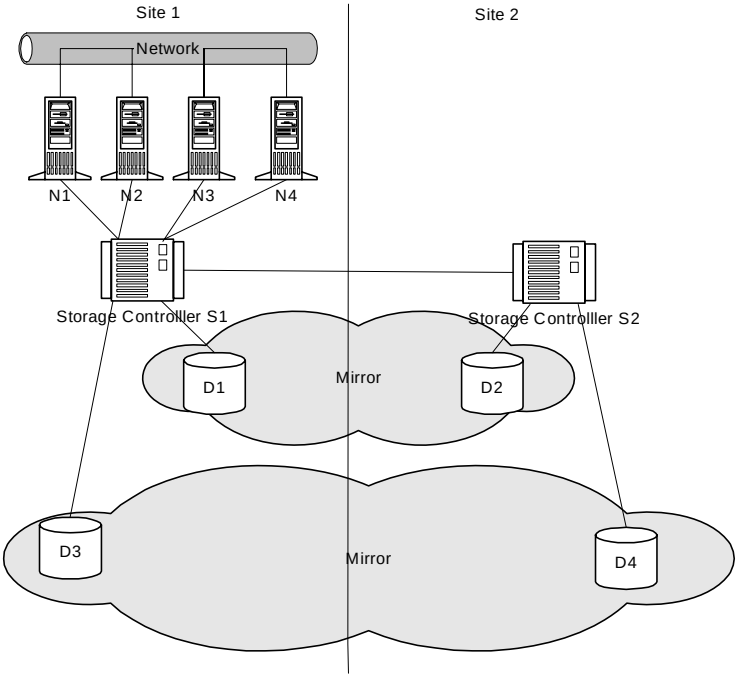


Figure 3. Disaster recovery solution without using geographically dispersed clusters

In this example, all the nodes in the cluster are at a single site. Data is mirrored to a remote site for disaster recovery purposes. This can be used in a number of different ways:

- Data vaulting – automatically copying data and updates to an off-site location to provide an off-site backup.
- Standby disaster recovery – There may be additional nodes at the secondary site that are not part of the cluster and may not be connected to the storage in the normal state. In the event of a disaster at the active site, nodes may be connected to the storage at the secondary site and the applications started (or even installed) as required, manually.

Data Replication

Data can be replicated using many different techniques at many different levels

- Block level – disk device level replication or mirroring. This is typically provided either by the storage controllers or by mirroring host software.
- File system level – replication of file system changes. This is typically provided by host software.
- Application level – Application specific replication mechanisms such as SQL Server log shipping.

How data should be replicated in a given scenario depends on the requirements of the application and the business. As well as understanding what level replication occurs at, you must also understand whether the data is replicated *synchronously* or *asynchronously*.

Synchronous replication means that if an application performs an operation on one node at one site, then that operation will not complete until the change has been made on the other sites. Consider the case of synchronous, block level replication. If an application at site A writes a block of data to a disk mirrored to site B, then the IO operation will not complete until the change has been made to the disk on site A AND the disk on site B.

Asynchronous replication means that if a change is made to the data on site A, that change will eventually make it to site B. Taking the same example as above, if an application at site A writes a block of data to a disk mirrored to site B, then the IO operation will complete as soon as the change is made to the disk at site A. The replication software will transfer the change to site B in the background and will eventually make that change to site B. Using asynchronous replication; the data at site B may be out of date with respect to site A at any point in time. Different vendors implement asynchronous replication in different ways. Some preserve the order of operations, others do not. If a solution preserves ordering, then the disk at site B may be out of date, but it will always represent a state that existed at site A at some point in the past. In other words, site B is *crash consistent*; the data at site B represents the data at site A if site A had crashed at that point in time. If a solution does not preserve ordering, the I/Os may be applied at site B in an arbitrary order. In this case, the data set at site B may never have existed at site A. Many applications can recover from crash consistent states; very few (if any) can recover from out of order I/O sequences. In short, never use asynchronous replication unless the order is preserved. If order is not preserved, the data on site B may well appear corrupt to the application and may be totally unusable.

Different mirroring and replication solutions are implemented differently. However, for each disk, there is typically a master and one or more secondary copies. The master can be modified and the changes are propagated to the secondary. In the case of device level replication, the secondary may or may not be visible to the applications. If it is visible, it will be a read-only copy of the device. In the event of a failover, a cluster resource typically switches one of the secondary copies to be the primary and the old primary becomes a secondary. In other words, most of the mirroring solutions are master-slave, one-way mirror sets. This is usually on a per-disk basis, so some disks may have the master at one site and others may have the master at another site.

Application Failover

Applications in a multi-site cluster are typically setup to failover just like a single-site cluster. MSCS itself provides health monitoring and failure detection of the applications, the nodes and the communications links. There are, however, cases where the software cannot differentiate between different failure modes.

The MSCS architecture requires there to be a single quorum resource in the cluster that is used as the tie-breaker to avoid split-brain scenarios. A split-brain scenario happens when all of the network communication links between two or more cluster nodes fail. In these cases, the cluster may be split into two or more partitions[1] that cannot communicate with each other. Each partition cannot communicate with the other partition(s) and cannot therefore differentiate the two cases:

[1] A partition is defined as a set of cluster nodes that can communicate with each other.

- Communication between sites failed and the other site is still alive
- The other site is dead and no long available to run applications

While this can certainly happen in a single-site cluster deployment, it is much more likely to happen in a multi-site configuration. The cluster service guarantees, that even in these cases, a resource is only brought online on one node (actually the guarantee is that it will never be brought online on more than one node). If the different partitions of the cluster each brought a given resource online, then it would violate the cluster guarantees and potentially cause data corruption. When the cluster is partitioned, the quorum resource is used as an arbiter; the partition that owns[2] the quorum resource is allowed to continue, the other partitions of the cluster are said to have lost quorum. The cluster service and any resources hosted on the nodes which were not part of the partition that has quorum are terminated.

The quorum resource is a storage-class resource and, in addition to being the arbiter in a split-brain scenario, it is used to store the definitive version of the cluster configuration. To ensure that the cluster always has an up-to-date copy of the latest configuration information, the quorum resource must itself be highly available. In Windows 2000, the quorum device was typically a shared disk or physical disk resource type.[3]

There are, however, cases where the software cannot make a decision about which site should host resources. Consider two identical sites, the same number of nodes and the same software installed. If there is a complete failure of all communication (both network and storage fabric) between the sites, neither site can decide to continue without manual intervention since neither has enough information to know whether the other site will continue or not. Different vendors solve this problem in different ways; however, they all require some form of administrator intervention to select which site should continue. The goal of any geographically dispersed configuration is to reduce the number of scenarios where manual intervention is required.

Some operational procedures require that manual intervention is always required in the event of a site loss. Typically, losing a site can mean that other procedures have to be initiated such as redirecting phones, moving personnel etc. Getting the applications up and running is a piece of a more complex puzzle that needs to be orchestrated within the business procedures.

[2] How a partition becomes the owner of the quorum resource is beyond the scope of this discussion.

[3] This is the only resource type supported as a quorum resource in the product shipped as part of Windows 2000. Other vendors have supplied alternative quorum resource types, but they are still typically associated with disks on a shared storage bus.

Deploying Geographically Dispersed Clusters

A geographically dispersed cluster is a combination of hardware and software. Microsoft does not provide a complete end-to-end geographically dispersed cluster solution; instead the MSCS feature set can be augmented and deployed on multi-site hardware configurations by OEMs and software vendors to allow an MSCS cluster to span multiple sites. In other words, a geographically dispersed cluster is a combination of pieces supplied by different vendors. Due to the complex nature of these configurations and the configuration restrictions that are fundamental to the MSCS technology, geographically dispersed clusters should be deployed only in conjunction with vendors who provide qualified configurations.

Qualified Configurations

As with any other MSCS configuration, the *complete solution* must be qualified and appear on the Microsoft Hardware Compatibility List (HCL) to be supported. The Windows 2000 HCL contains the following entries specifically for multi-site configurations:

Qualification	HCL Entry
Windows 2000 Advanced Server	Cluster/Geographic/2-node Advanced Server
Two-node Windows 2000 Datacenter Server	Cluster/Geographic/2-node Datacenter Server
Four-node Windows 2000 Datacenter Server	Cluster/Geographic/4-node Datacenter Server

Microsoft supports only configurations that appear on the lists at http://www.microsoft.com/hcl.

Geographically dispersed cluster configurations are very complex and the hardware and system software must provide very strict guarantees, especially around storage and around how split-brain scenarios are handled. There are many subtle issues that are not immediately obvious in these configurations that will cause loss of data or data corruption. The Microsoft Geographic Cluster qualification program that allows vendors to list solutions on the HCL performs rigorous checks for various failure cases to ensure data integrity and to ensure that the cluster guarantees are always met.

MSCS Technology Restrictions

The Microsoft server clustering software itself is unaware of the extended nature of geographically dispersed clusters. There are no special features in MSCS in Windows 2000 that are specific to these kinds of configuration. The network and storage architectures used to build geographically dispersed clusters must preserve the semantics that the Server cluster technology expects. Fundamentally, the network and storage architecture of geographically dispersed Server clusters must meet the following requirements:

1. The private and public network connections between cluster nodes must appear as a single, non-routed LAN (e.g., using technologies such as VLANs to ensure that all nodes in the cluster appear on the same IP subnets). This is essential for a number of reasons including:

a. To detect node failures, NIC failures and failures of network infrastructure, MSCS implement a network topology detection mechanism. This mechanism makes many simplifying assumptions about the configuration that are not possible in a complex network topology containing routers, switches etc.

b. MSCS provides IP address failover to allow clients to access application seamlessly even though the application is now running on the other node. This allows legacy clients to connect to clustered services as the IP address and network name to IP address translation never changes regardless of where the application is running. IP address failover cannot be used effectively in a cluster that spans multiple subnets. Although it would be possible to inject routing information, the routing protocols take a significant time to become consistent. During that time, the client may or may not have access to the service. It would also be possible to allow a service to change its IP address as it fails over between cluster nodes, however, that would require changes to the different clients of services running on the cluster.

2. The network connections must be able to provide a *maximum guaranteed* round trip latency between nodes of no more than 500 milliseconds. The cluster uses heartbeat to detect whether a node is alive or not responding. These heartbeats are sent out on a periodic basis (every 1.2 seconds). If a node takes too long to respond to heartbeat packets, MSCS starts a heavy-weight protocol to figure out which nodes are really still alive and which ones are dead; this is known as a cluster re-group. The heartbeat interval is not a configurable parameter for the cluster service (there are many reasons for this, but the bottom line is that changing this parameter can have a significant impact on the stability of the cluster and the failover time). 500 ms round-trip is significantly below any threshold to ensure that artificial re-group operations are not triggered.

3. Windows 2000 requires that a cluster have a single shared disk known as the quorum disk[4]. The storage infrastructure can provide mirroring across the sites to make a set of disks appear to MSCS like a single disk, however, it must preserve the fundamental semantics that are required by the physical disk resource:

   a. Cluster service uses SCSI reserve commands and bus reset to arbitrate for and protect the shared disks. The semantics of these commands must be preserved across the sites, even in the face of complete communication failures between sites. If a node on site A reserves a disk, nodes on site B should not be able to access the contents of the disk. These semantics are essential to avoid data corruption of cluster data and application data.

   b. The quorum disk must be replicated in real-time, synchronous mode across all sites. The different members of a mirrored quorum disk MUST contain the same data.

Other Considerations

Geographically dispersed clusters are complex and you should fully understand the proposed solution and the operational procedures as part of the design process. This section gives you some things to think about (and things to ask the vendor about) as part of the planning process.

---

4 Vendors can provide a quorum capable resource rather than try to emulate a physical disk that spans multiple sites. The MSCS SDK presents the requirements for building these types of resource. Many of the vendors, however, are storage vendors and they have chosen to implement multi-site mirroring at the storage level and use the physical disk resource (e.g. EMC with the GeoSpan solution).

Basic Architecture

1. How is data on cluster disks (disks on a shared bus connected to cluster nodes) mirrored between sites? Is it at the block level or the file system level?
2. Is mirroring synchronous or asynchronous? (If an application does an IO to the mirror, when is the application told that the IO completed with respect to the data being on the disks at each site)?
3. Do nodes at both sites have visibility to a cluster disk at the same time?
4. Is the mirroring multi-master (a write at either side will be mirrored to both sites) or is it primary-secondary (only one site can modify the data at any one time)?
5. In a primary-secondary mirroring configuration, what mechanisms are used to switch roles in the event of a failover?
6. If nodes at both sites have visibility to a cluster disk at the same time, how do you stop both sites from accessing the different sides of the mirror in the event of a total communication failure between sites?
7. How do you propose to ensure that the latency of disk operations is within the bounds required to support applications such as Microsoft SQL Server and Microsoft Exchange Server? How will you ensure that these types of applications work as expected in this environment?
8. Apart from disk mirroring, do you provide any other disk virtualization mechanism?
9. What do you use as a quorum resource in a geographically dispersed Server cluster?
10. Give a brief description of the quorum mechanism that you are using.
11. In the event of a total communication failure between sites, how do you ensure that only one site continues, i.e., how do you avoid split-brain scenarios?
12. What are the possible network topologies for both the client access (public) network and the intracluster (private) network?
13. What are the possible VLAN technologies in use and how are they configured to ensure that all nodes in the cluster are in a single subnet?
14. How do you guarantee that the round-trip latency of the private network is less than 500 ms between sites?

Operational and Deployment Procedures

1. How do you ensure that the deployment at the customer site is within the limits required by MSCS (<500 ms roundtrip latency on the network interconnect)?
2. What procedures do you require a customer to implement when changing such things as inter-site network suppliers and/or technologies?
3. What is the procedure by which a customer upgrades firmware or driver versions on HBAs, Storage Controllers, active components in the network or storage fabric, etc.?
4. Are the geographically dispersed Server clusters able to run in active/active mode, i.e., with both sites servicing requests for applications?

5. What mechanisms do you describe to the administrators to allow them to set up a geographically dispersed Server cluster in which an application fails over within a site before any cross-site failover is initiated?
6. In the event of a total communications failure between sites, what is the operational procedure that is required to continue servicing client requests? What is the protection that ensures both sites cannot be up and running at the same time?
7. In a two-site configuration with both a primary and a secondary site (where the primary site is defined as the site that currently owns the quorum resource), what is the procedure to allow the primary site to continue in the event that the secondary site goes down? What is the procedure to allow the secondary site to continue in the event that the primary site goes down?
8. Do the application specific resources (such as Microsoft SQL Server, Microsoft Exchange Server, etc.) need fine-tuning of the failure detection poll-intervals (LooksAlive, IsAlive) to prevent false failures from being detected due to increased latency of operations?

Server Cluster Parameters

Server cluster provides parameters that can be used to tune the re-group or arbitration time in a geographically dispersed cluster. In a geographically dispersed cluster, when a re-group happens due to a failure or a recovery from a failure, the storage components typically have to re-synchronize themselves. Consider the case where the link between two sites failed and a re-group happens. When the link comes back up, the mirror sets have to be re-synchronized at both sites to ensure that they have the same, consistent view of the data. This can involve a delay while data is shipped between sites. This re-synchronization needs to happen before the cluster re-arbitrates since one of the disks may in fact be the quorum disk, therefore, in a geographically dispersed cluster, arbitration may take longer.

The parameters that affect arbitration are:

1. MinArbitrationTime - If arbitration on a node takes less than MinArbitrationTime, MSCS will pad the arbitration time to be at least MinArbitrationTime. The default value is 7 seconds.
2. MaxArbitrationTime - If arbitration on a node takes longer than MaxArbitrationTime, that node will be killed and a new node will try to arbitrate. The default value is 60 seconds.

The re-group engine has some unconfigurable limits on how quickly nodes have to respond to a re-group before they are considered to be dead. For a node to be considered alive during re-group, it must respond to the re-group packet in less than four seconds.

Applications on Multi-site Clusters

Applications such as SQL Server and Exchange can be deployed on a geographically dispersed cluster without change. As with MSCS itself, Exchange and SQL Server are oblivious to the fact that the cluster, in fact, spans nodes.

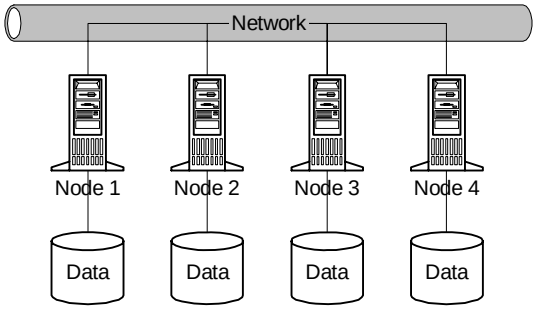
> *Note: In some cases you may need to change the IsAlive and LooksAlive polling interval parameters for the cluster resources. This is due to the increased latency of IO operations (especially if data is synchronously mirrored across the sites) and other operations that need to span sites.*

As a default statement, the cluster team recommends that you use synchronous mirroring of data disks between sites. This ensures the correct behavior of applications in the event of failover across sites (i.e., the data is up-to-date on all sites). In some cases, it may be possible to use asynchronous replication; however, this depends on the semantics provided by the mirroring/replication solution and the application requirements. For example, SQL Server requires (and makes assumptions about) the fact that I/Os are done in order. If the asynchronous mirroring solution does not complete I/Os on the secondary site in the same order as SQL Server performed them on the primary site, this will lead to corrupt databases. Even if the I/Os are done in order, in the event of failover to the secondary site, the data may not be up-to-date. This may break the requirements of the client application. You should always understand the business requirements of the solution. In the event of failures, refer to the application documentation and work closely with the vendors to understand the implications of using asynchronous replication or mirroring.

Server Cluster Features in Windows Server 2003

Future versions of Windows cluster technologies will better support features that are applicable to geographically dispersed clusters. The Windows Server 2003 family takes the first few steps along that path by providing the following features that enable more flexible geographically dispersed cluster topologies:

- 8-Node Clusters – The Windows Server 2003 family allows 8-node clusters. This allows more flexibility in the way multi-site clusters can be managed. For example, with four nodes at each site, rolling upgrades of either the OS or application software is possible without needing to failover between sites if another node fails while rolling upgrade is in progress.
- Majority node set quorum – A majority node set is a single quorum resource from an MSCS perspective; however, the data is actually stored on multiple disks across the cluster. The majority node set resource takes care to ensure that the cluster configuration data stored on the majority node set is kept consistent across the different disks. This allows cluster topologies as follows:



The disks that make up the majority node set could, in principle, be local disks physically attached to the nodes themselves or disks on a shared storage fabric. In the majority node set implementation that is provided as part of MSCS in Windows Server 2003, every node in the cluster uses a directory on its own local system disk to store the quorum data. If the configuration of the cluster changes, that change is reflected across the different disks. The change is only considered to have been committed (i.e. made persistent), if that change is made to:

(<Number of nodes configured in the cluster>/2) + 1

This ensures that a majority of the nodes have an up-to-date copy of the data. The cluster service itself will only start up, and therefore bring resources on line, if a majority of the nodes configured as part of the cluster are up and running the cluster service. If there are fewer nodes, the cluster is said not to have quorum and therefore the cluster service waits (trying to restart) until more nodes try to join. Only when a majority or quorum of nodes, are available, will the cluster service start up and bring the resources online. This way, since the up-to-date configuration is written to a majority of the nodes regardless of node failures, the cluster will always guarantee that it starts up with the latest and most up-to-date configuration.

In the case of a failure or split-brain, all resources hosted on the partition of a cluster that has lost quorum are terminated to preserve the cluster guarantees of only bringing a resource online on one node. If a cluster becomes partitioned (i.e. there is a communications failure between two sets of nodes in the cluster), then any partitions that do not have a majority of the configured nodes (i.e. have less than (n/2)+1 nodes) are said to have lost quorum. The cluster service and all resources hosted on the nodes of partitions that do not have a majority are terminated. This ensures that if there is a partition running that contains a majority of the nodes, it can safely start up any resources that are not running on that partition, safe in the knowledge that it can be the only partition in the cluster that is running resources (since all other partitions must have lost quorum).

By providing this optional quorum mechanism, geographically dispersed clusters can use a Microsoft supplied quorum mechanism that does not require a shared disk to span the sites. For some solutions, where data is replicated above the device level, such as the file system level (e.g. file system replication) or at the application level (e.g. database log shipping), this quorum resource removes any need for a shared storage fabric to span multiple sites.

For more information about the majority node set quorum resource see *Server Clusters: Majority Node Sets at* http://www.microsoft.com/windows2000/technologies/clustering/default.asp

Three-Site Majority Node Set Quorum in Geographically Dispersed Clusters to Facilitate Automatic Failover

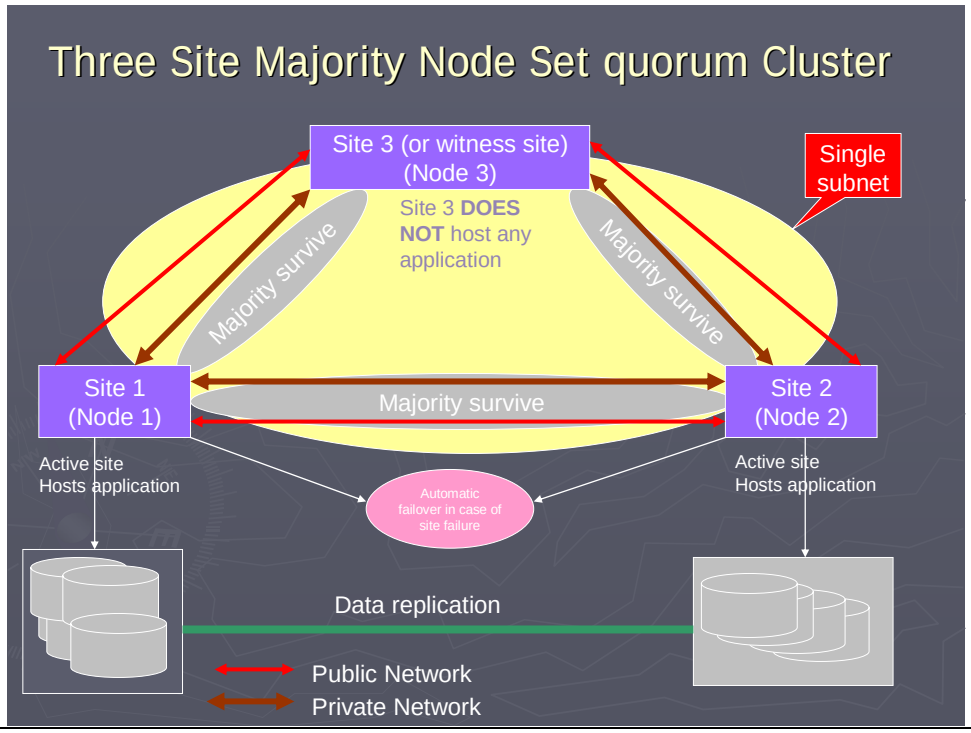
As mentioned in the 'Application Failover' section above, applications in a multi-site cluster are typically set up to fail over just like a single-site cluster. MSCS itself provides health monitoring and failure detection of the applications, the nodes and the communications links. There are, however, cases where the software cannot differentiate between different failure modes.

Consider two identical sites, each having the same number of nodes and running the same software. If a complete failure of all communication (both network interfaces and storage fabric) occurs between the sites, neither site can continue without human intervention or an arbitrator because neither site has sufficient information about the other site's continuity. This scenario is commonly referred to as split-brain.

Geographically dispersed clusters configured with Majority Node Set (MNS) quorum can be deployed and configured in a way that will automate the failover of applications in situations where the following occurs

- Communication between sites has failed and the one site is still functioning
- The other site is down and no long available to run applications

In this type of configuration the nodes of the cluster are dispersed among three different sites, where the third site acts as 'Quorum site' for cluster to maintain node majority. The quorum site node's sole purpose is to allow the cluster to retain majority of the nodes for the cluster to continue operation in case of total site failure. For instance, in a three-node cluster, one node each is distributed among the three sites. In case of a complete site failure, the cluster will continue to run since the nodes on the two available sites can form a majority. In this scenario, the applications are automatically failed over to the surviving site. The figure below illustrates the three-site concept.



In the figure above, a three-node three-site Majority Node Set (MNS) quorum cluster is illustrated. Site 1 and site 2 host the application and have shared storage connected to it. The 3rd site doesn't host any application or storage. The application resources are configured never to failover to site 3, by appropriate settings of the "Possible Owner" parameter. The 3rd site node provides majority to cluster algorithm when there is a complete failure at site 1 or site 2. As illustrated, this configuration allows the cluster to survive one complete site failure. Please refer to Majority Node Set quorum mechanism white paper on www.microsoft.com for more information.

Following are configuration & deployment requirements to get the three-site MNS cluster configured and supported by Microsoft.

1. The private and public network connections between cluster nodes must appear as a single, non-routed LAN (for example, using technologies such as VLANs to ensure that all nodes in the cluster appear on the same IP subnets) between the three sites.
    a. All three-site public interface must be on same subnet
    b. All three-site private interface must be on same subnet
2. The network connections between the sites must be able to provide a *maximum guaranteed* round trip latency of no more than 500 milliseconds.
3. The cluster can have shared disk only on two sites. The storage infrastructure can provide mirroring across these 2 sites to meet the application requirements. The shared storage and storage replication should not be extended to the third site.
4. The third site acts as a 'Quorum' site only to retain a majority of nodes in case of a site failure. This can be done by NOT installing the application on the third node, and by removing the third site node from the each resource of the application resources group possible owner list.
    a. Must not host any application, this can be achieved by removing the node from possible owner list of each resource of the group.
    b. Must not host any application data
    c. Must not be attached to shared disk
    d. Only one node should be in the third site
    e. Application resources must never failover to third site
    f. ONLY the cluster group, which contains the MNS resource must be able to failover to this third site node. Cluster group should not contain any application specific resources.
5. The node in the third site can be any machine that is capable of running Windows Server 2003 Enterprise, or Datacenter Edition.
6. Windows catalog will not list three-site majority node set based geographically dispersed cluster separately
    a. Two-site majority node set based geographically dispersed cluster and a three-site majority node set based geographically dispersed clusters are considered one and same in terms of windows catalog listing as along as the above requirements are followed.

With a three site cluster, potentially one less node will be available to host the application. For example, in case of a five node cluster, four nodes will be available to host the application. The fifth node will be in the third site to help retain majority. To alleviate the cost issue, the system deployed on the third site can be any Windows Server 2003, Enterprise Edition, capable system.

As discussed in the 'Other Considerations' section of this white paper, before you deploy a three-site solution, you should consider your business requirements. Is automatic failover of applications and immediately functioning cluster and application in case of total site failure a key requirement for your business?

Summary

Windows server cluster can be used to provide a disaster-tolerant environment for mission critical data and applications. This is essential considering that more and more enterprises are running applications that are fundamental to the core business. The Windows Server 2003 family meets the requirements for disaster tolerance and business continuance by ensuring that there is *no single point of failure.* This level of availability is achieved by protecting against storage, network, computer and site failures.

This paper discusses mult-site network load balancing and cluster service configurations, along with how to deploy geographically dispersed clusters. Also covered are key server cluster features in Windows Server 2003.

Related Links

Technical Overview of Clustering Services at
http://www.microsoft.com/windows.netserver/techinfo/overview/clustering.mspx

What's New in Clustering Technologies at
http://www.microsoft.com/windows.netserver/evaluation/overview/technologies/clustering.mspx

Clustering Technologies at
http://www.microsoft.com/windows2000/technologies/clustering/default.asp

Microsoft Hardware Compatibility List at http://www.microsoft.com/hcl

For the latest information about Windows Server 2003, see the Windows Server 2003 Web site at

For the latest information about Windows Server 2003, see the Windows Server 2003 Web site at http://www.microsoft.com/windowsserver2003.