Data Center High Availability Clusters

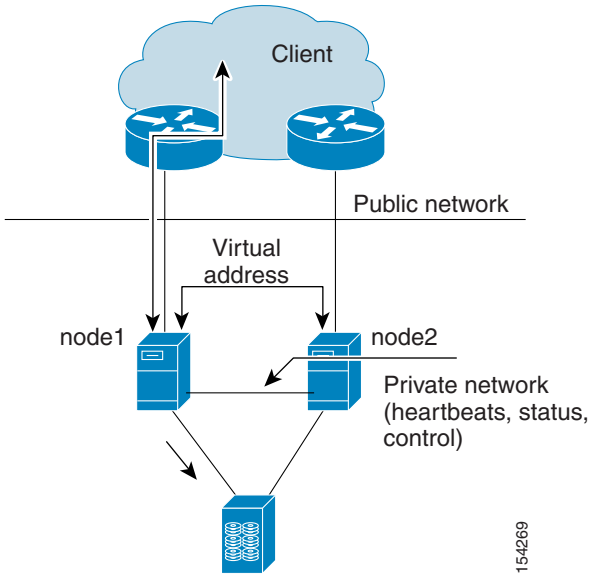
High Availability Clusters Overview

Clusters define a collection of servers that operate as if they were a single machine. The primary purpose of high availability (HA) clusters is to provide uninterrupted access to data, even if a server loses network or storage connectivity, or fails completely, or if the application running on the server fails.

HA clusters are mainly used for e-mail and database servers, and for file sharing. In their most basic implementation, HA clusters consist of two server machines (referred to as "nodes") that "share" common storage. Data is saved to this storage, and if one node cannot provide access to it, the other node can take client requests. Figure 1-1 shows a typical two node HA cluster with the servers connected to a shared storage (a disk array). During normal operation, only one server is processing client requests and has access to the storage; this may vary with different vendors, depending on the implementation of clustering.

HA clusters can be deployed in a server farm in a single physical facility, in different facilities at various distances for added resiliency. The latter type of cluster is often referred to as a *geocluster*.

Figure 1-1 Basic HA Cluster



Geoclusters are becoming very popular as a tool to implement business continuance. Geoclusters improve the time that it takes for an application to be brought online after the servers in the primary site become unavailable. In business continuance terminology, geoclusters combine with disk-based replication to offer better recovery time objective (RTO) than tape restore or manual migration.

HA clusters can be categorized according to various parameters, such as the following:

- How hardware is shared (shared nothing, shared disk, shared everything)
- At which level the system is clustered (OS level clustering, application level clustering)
- Applications that can be clustered
- Quorum approach
- Interconnect required

One of the most relevant ways to categorize HA clusters is how hardware is shared, and more specifically, how storage is shared. There are three main cluster categories:

- Clusters using mirrored disks—Volume manager software is used to create mirrored disks across all the machines in the cluster. Each server writes to the disks that it owns and to the disks of the other servers that are part of the same cluster.
- Shared nothing clusters—At any given time, only one node owns a disk. When a node fails, another node in the cluster has access to the same disk. Typical examples include IBM High Availability Cluster Multiprocessing (HACMP) and Microsoft Cluster Server (MSCS).
- Shared disk—All nodes have access to the same storage. A locking mechanism protects against race conditions and data corruption. Typical examples include IBM Mainframe Sysplex technology and Oracle Real Application Cluster.

Technologies that may be required to implement shared disk clusters include a *distributed volume manager,* which is used to virtualize the underlying storage for all servers to access the same storage; and the *cluster file system,* which controls read/write access to a single file system on the shared SAN.

More sophisticated clustering technologies offer shared-everything capabilities, where not only the file system is shared, but memory and processors, thus offering to the user a *single system image (SSI)*. In this model, applications do not need to be cluster-aware. Processes are launched on any of the available processors, and if a server/processor becomes unavailable, the process is restarted on a different processor.

The following list provides a partial list of clustering software from various vendors, including the architecture to which it belongs, the operating system on which it runs, and which application it can support:

- HP MC/Serviceguard—Clustering software for HP-UX (the OS running on HP Integrity servers and PA-RISC platforms) and Linux. HP Serviceguard on HP-UX provides clustering for Oracle, Informix, Sybase, DB2, Progress, NFS, Apache, and Tomcat. HP Serviceguard on Linux provides clustering for Apache, NFS, MySQL, Oracle, Samba, PostgreSQL, Tomcat, and SendMail. For more information, see the following URL: http://h71028.www7.hp.com/enterprise/cache/4189-0-0-0-121.html.
- HP NonStop computing—Provides clusters that run with the HP NonStop OS. NonStop OS runs on the HP Integrity line of servers (which uses Intel Itanium processors) and the NonStop S-series servers (which use MIPS processors). NonStop uses a shared nothing architecture and was developed by Tandem Computers. For more information, see the following URL: http://h20223.www2.hp.com/nonstopcomputing/cache/76385-0-0-0-121.aspx
- HP OpenVMS High Availability Cluster Service—This clustering solution was originally developed for VAX systems, and now runs on HP Alpha and HP Integrity servers. This is an OS-level clustering that offers an SSI. For more information, see the following URL: http://h71000.www7.hp.com/.

- HP TruCluster—Clusters for Tru64 UNIX (aka Digital UNIX). Tru64 Unix runs on HP Alpha servers. This is an OS-level clustering that offers an SSI. For more information, see the following URL: http://h30097.www3.hp.com/cluster/
- IBM HACMP—Clustering software for servers running AIX and Linux. HACMP is based on a shared nothing architecture. For more information, see the following URL: http://www-03.ibm.com/systems/p/software/hacmp.html
- MSCS—Belongs to the category of clusters that are referred to as shared nothing. MSCS can provide clustering for applications such as file shares, Microsoft SQL databases, and Exchange servers. For more information, see the following URL: http://www.microsoft.com/windowsserver2003/technologies/clustering/default.mspx
- Oracle Real Application Cluster (RAC) provides a shared disk solution that runs on Solaris, HP-UX, Windows, HP Tru64 UNIX, Linux, AIX, and OS/390. For more information about Oracle RAC 10g, see the following URL: http://www.oracle.com/technology/products/database/clustering/index.html
- Solaris SUN Cluster—Runs on Solaris and supports many applications including Oracle, Siebel, SAP, and Sybase. For more information, see the following URL: http://wwws.sun.com/software/cluster/index.html
- Veritas (now Symantec) Cluster Server—Veritas is a "mirrored disk" cluster. Veritas supports applications such as Microsoft Exchange, Microsoft SQL Databases, SAP, BEA, Siebel, Oracle, DB2, Peoplesoft, and Sybase. In addition to these applications you can create agents to support custom applications. It runs on HP-UX, Solaris, Windows, AIX, and Linux. For more information, see the following URL: http://www.veritas.com/us/products/clusterserver/prodinfo.html and http://www.veritas.com/Products/www?c=product&refId=20.

Note A single server can run several server clustering software packages to provide high availability for different server resources.

Note For more information about the performance of database clusters, see the following URL: http://www.tpc.org

Clusters can be "stretched" to distances beyond the local data center facility to provide metro or regional clusters. Virtually any cluster software can be configured to run as a *stretch cluster*, which means a cluster at metro distances. Vendors of cluster software often offer a geoclusters version of their software that has been specifically designed to have no intrinsic distance limitations. Examples of geoclustering software include the following:

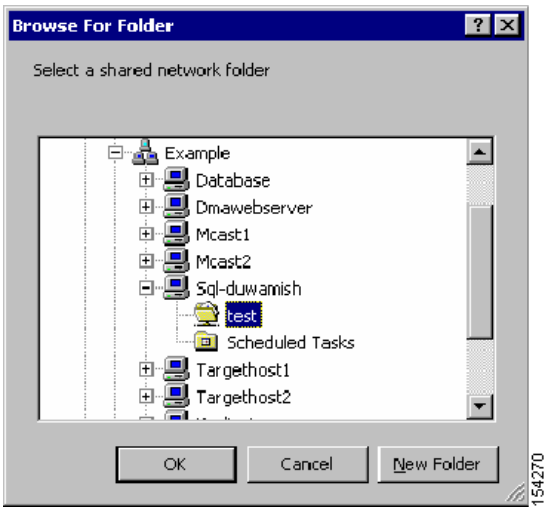
- EMC Automated Availability Manager Data Source (also called AAM)—This HA clustering solution can be used for both local and geographical clusters. It supports Solaris, HP-UX, AIX, Linux, and Windows. AAM supports several applications including Oracle, Exchange, SQL Server, and Windows services. It supports a wide variety of file systems and volume managers. AAM supports EMC SRDF/S and SRDF/A storage-based replication solutions. For more information, see the following URL: http://www.legato.com/products/autostart/
- Oracle Data Guard—Provides data protection for databases situated at data centers at metro, regional, or even continental distances. It is based on redo log shipping between active and standby databases. For more information, see the following URL: http://www.oracle.com/technology/deploy/availability/htdocs/DataGuardOverview.html
- Veritas (now Symantec) Global Cluster Manager—Allows failover from local clusters in one site to a local cluster in a remote site. It runs on Solaris, HP-UX, and Windows. For more information, see the following URL: http://www.veritas.com/us/products/gcmanager/

- HP Continental Cluster for HP-UX—For more information, see the following URL: http://docs.hp.com/en/B7660-90013/index.html
- IBM HACMP/XD (Extended Distance)—Available with various data replication technology combinations such as HACMP/XD Geographic Logical Volume Manager (GLVM) and HACMP/XD HAGEO replication for geographical distances. For more information, see the following URL: http://www-03.ibm.com/servers/systems/p/ha/disaster_tech.html

HA Clusters Basics

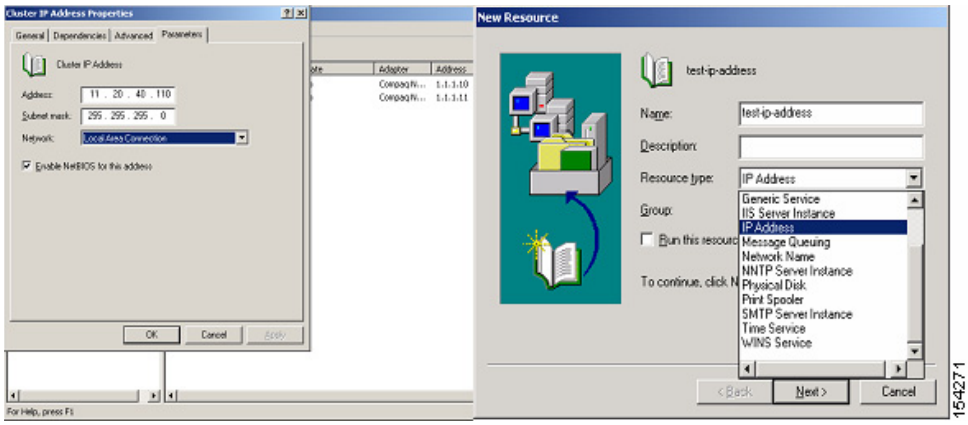
HA clusters are typically made of two servers such as the configuration shown in Figure 1-1. One server is actively processing client requests, while the other server is monitoring the main server to take over if the primary one fails. When the cluster consists of two servers, the monitoring can happen on a dedicated cable that interconnects the two machines, or on the network. From a client point of view, the application is accessible via a name (for example, a DNS name), which in turn maps to a virtual IP address that can float from a machine to another, depending on which machine is active. Figure 1-2 shows a clustered file-share.

Figure 1-2 Client Access to a Clustered Application—File Share Example



In this example, the client sends requests to the machine named “sql-duwamish”, whose IP address is a virtual address, which could be owned by either node1 or node2. The left of Figure 1-3 shows the configuration of a cluster IP address. From the clustering software point of view, this IP address appears as a monitored resource and is tied to the application, as described in Concept of Group, page 1-7. In this case, the IP address for the “sql-duwamish” is 11.20.40.110, and is associated with the clustered application “shared folder” called “test”.

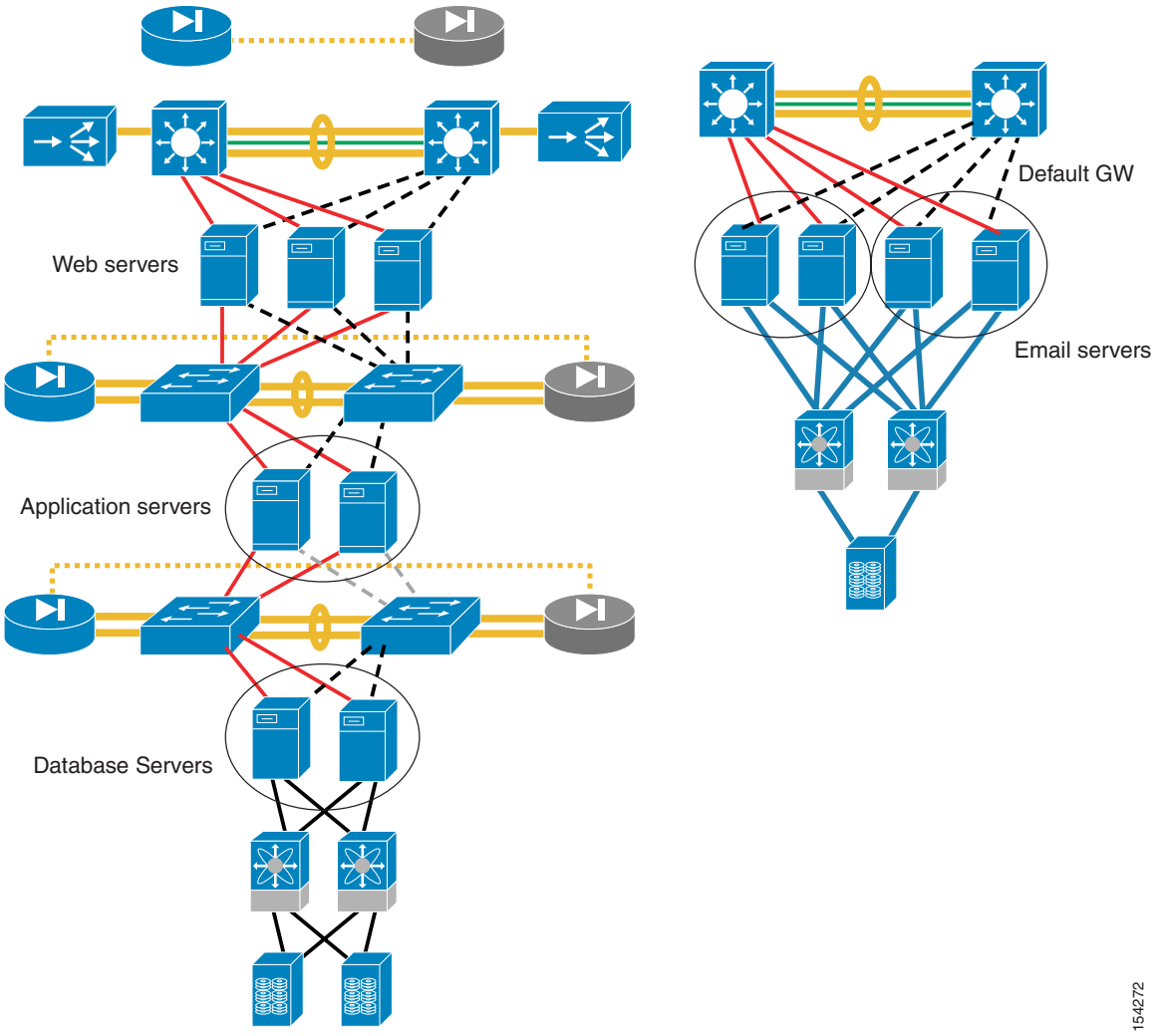
Figure 1-3 Virtual Address Configuration with MSCS



HA Clusters in Server Farms

Figure 1-4 shows where HA clusters are typically deployed in a server farm. Databases are typically clustered to appear as a single machine to the upstream web/application servers. In multi-tier applications such as a J2EE based-application and Microsoft .NET, this type of cluster is used at the very bottom of the processing tiers to protect application data.

Figure 1-4 HA Clusters Use in Typical Server Farms



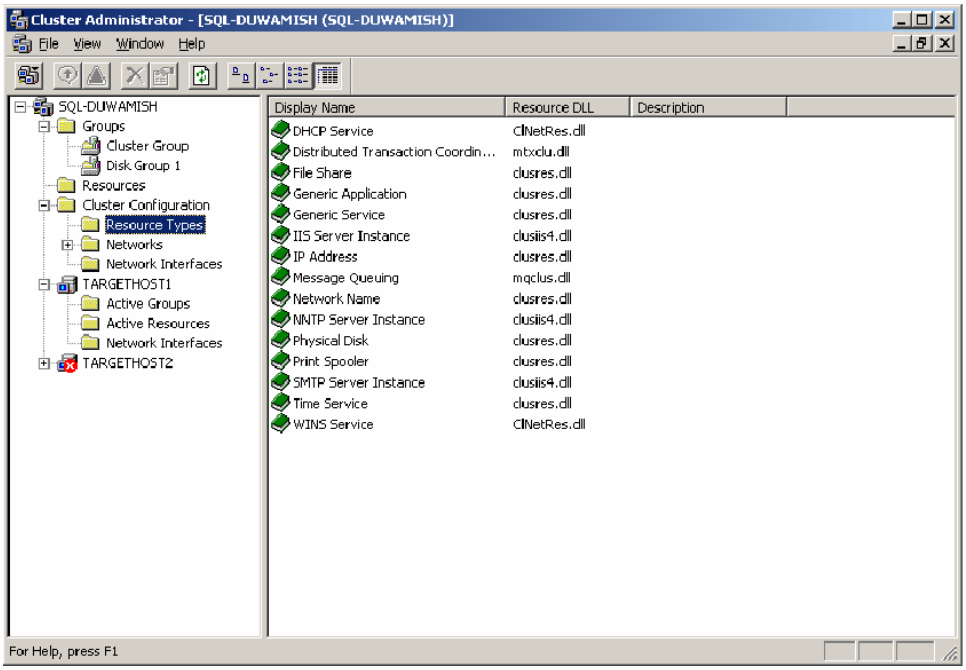
Applications

An application running on a server that has clustering software installed does not mean that the application is going to benefit from the clustering. Unless an application is cluster-aware, an application process crashing does not necessarily cause a failover to the process running on the redundant machine. Similarly, if the public network interface card (NIC) of the main machine fails, there is no guarantee that the application processing will fail over to the redundant server. For this to happen, you need an application that is cluster-aware.

Each vendor of cluster software provides immediate support for certain applications. For example, Veritas provides enterprise agents for the SQL Server and Exchange, among others. You can also develop your own agent for other applications. Similarly, EMC AAM provides application modules for Oracle, Exchange, SQL Server, and so forth.

In the case of MSCS, the cluster service monitors all the resources by means of the Resource Manager, which monitors the state of the application via the "Application DLL". By default, MSCS provides support for several application types, as shown in Figure 1-5. For example, MSCS monitors a clustered SQL database by means of the distributed transaction coordinator DLL.

Figure 1-5 Example of Resource DLL from MSCS

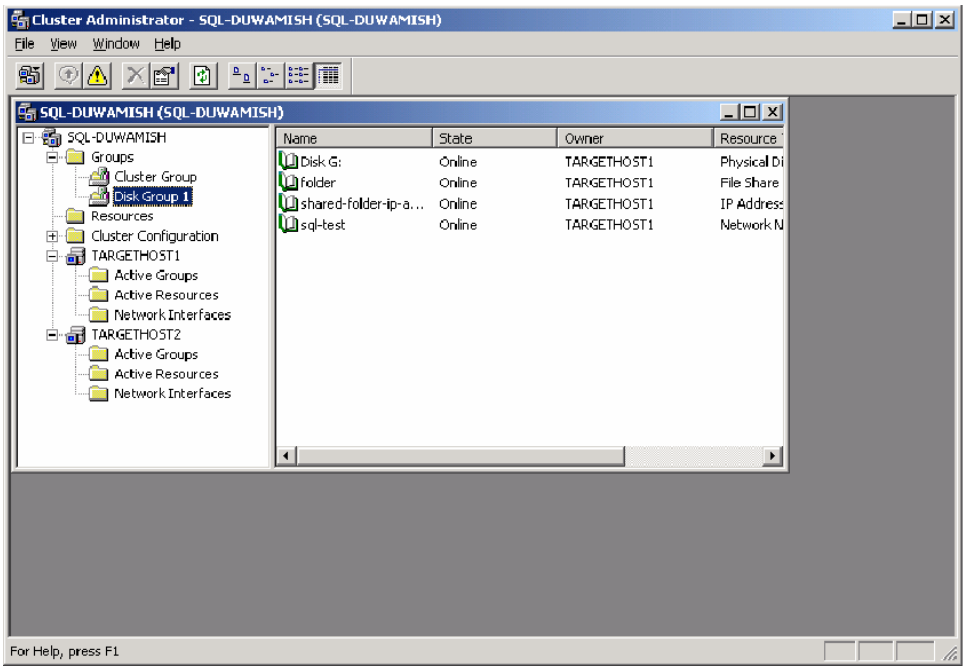


It is not uncommon for a server to run several clustering applications. For example, you can run one software program to cluster a particular database, another program to cluster the file system, and still another program to cluster a different application. It is out of the scope of this document to go into the details of this type of deployment, but it is important to realize that the network requirements of a clustered server might require considering not just one but multiple clustering software applications. For example, you can deploy MSCS to provide clustering for an SQL database, and you might also install EMC SRDF Cluster Enabler to failover the disks. The LAN communication profile of the MSCS software is different than the profile of the EMC SRDF CE software.

Concept of Group

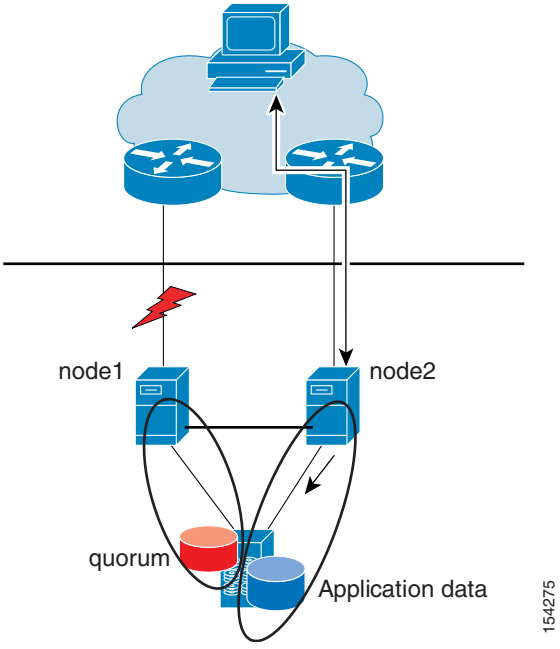
One key concept with clusters is the *group*. The group is a unit of failover; in other words, it is the bundling of all the resources that constitute an application, including its IP address, its name, the disks, and so on. Figure 1-6 shows an example of the grouping of resources: the "shared folder" application, its IP address, the disk that this application uses, and the network name. If any one of these resources is not available, for example if the disk is not reachable by this server, the group fails over to the redundant machine.

Figure 1-6 Example of Group



The failover of a group from one machine to another one can be automatic or manual. It happens automatically when a key resource in the group fails. Figure 1-7 shows an example: when the NIC on node1 goes down, the application group fails over to node2. This is shown by the fact that after the failover, node2 owns the disk that stores the application data. When a failover happens, node2 mounts the disk and starts the application by using the API provided by the Application DLL.

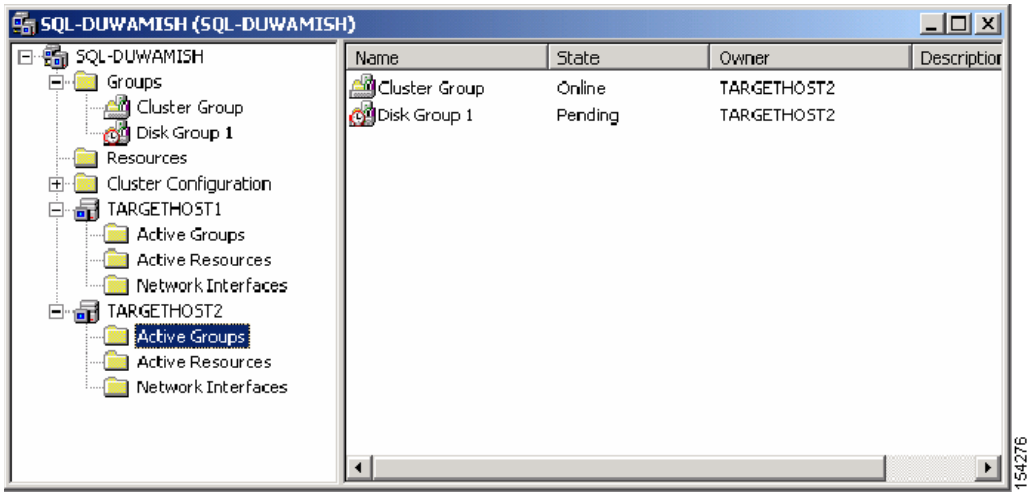
Figure 1-7 Failover of Group



The failover can also be manual, in which case it is called a *move*. Figure 1-8 shows a group (DiskGroup1) failing over to a node or "target2" (see the *owner* of the group), either as the result of a move or as the result of a failure.

After the failover or move, nothing changes from the client perspective. The only difference is that the machine that receives the traffic is node2 or target2, instead of node1 (or target1, as it is called in these examples).

Figure 1-8 Move of a Group



LAN Communication

The LAN communication between the nodes of a cluster obviously depends on the software vendor that provides the clustering function. As previously stated, to assess the network requirements, it is important to know all the various software components running on the server that are providing clustering functions.

Virtual IP Address

The virtual IP address (VIP) is the floating IP address associated with a given application or *group*. Figure 1-3 shows the VIP for the clustered shared folder (that is, DiskGroup1 in the group configuration). In this example, the VIP is 11.20.40.110. The physical address for node1 (or target1) could be 11.20.40.5, and the address for node2 could be 11.20.40.6. When the VIP and its associated group are active on node1, when traffic comes into the public network VLAN, either router uses ARP to determine the VIP and node1 answer. When the VIP *moves* or *fails over* to node2, then node2 answers the ARP requests from the routers.

Note From this description, it appears that the two nodes that form the cluster need to be part of the same subnet, because the VIP address stays the same after a failover. This is true for most clusters, except when they are geographically connected, in which case certain vendors allow solutions where the IP address can be different at each location, and the DNS resolution process takes care of mapping incoming requests to the new address.

The following trace helps explaining this concept:

```
11.20.40.6    11.20.40.1    ICMP      Echo (ping) request
11.20.40.1    11.20.40.6    ICMP      Echo (ping) reply
11.20.40.6    Broadcast     ARP       Who has 11.20.40.110?  Tell 11.20.40.6
11.20.40.6    Broadcast     ARP       Who has 11.20.40.110?  Gratuitous ARP
```

When 11.20.40.5 fails, 11.20.40.6 detects this by using the heartbeats, and then verifies its connectivity to 11.20.40.1. It then announces its MAC address, sending out a gratuitous ARP that indicates that 11.20.40.110 has moved to 11.20.40.6.

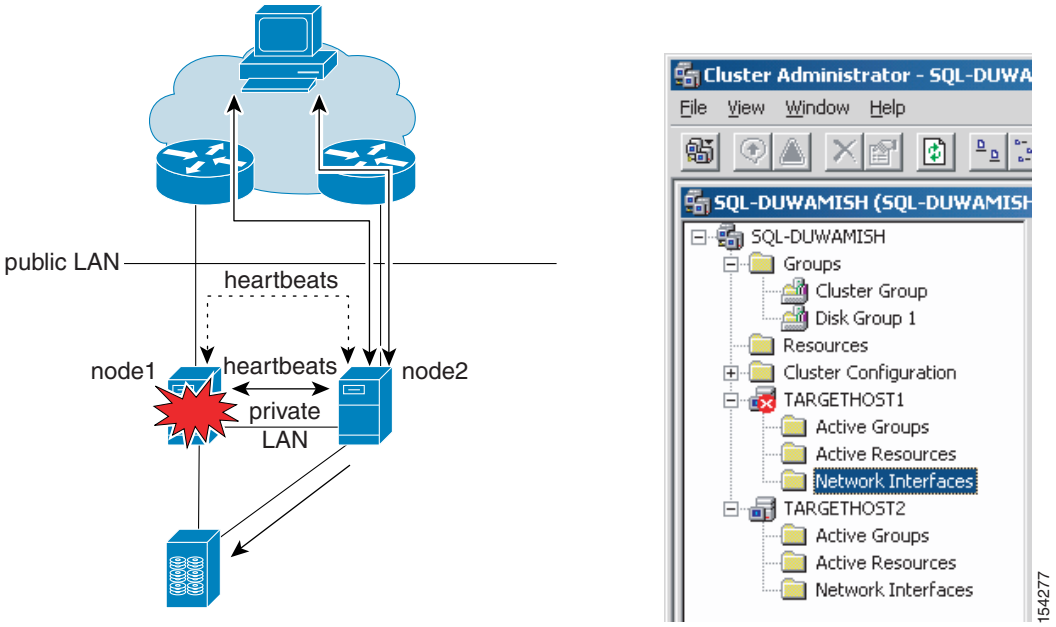
Public and Private Interface

As previously mentioned, the nodes in a cluster communicate over a public and a private network. The public network is used to receive client requests, while the private network is mainly used for monitoring. Node1 and node2 monitor the health of each other by exchanging heartbeats on the private network. If the private network becomes unavailable, they can use the public network. You can have more than one private network connection for redundancy. Figure 1-1 shows the public network, and a direct connection between the servers for the private network. Most deployments simply use a different VLAN for the private network connection.

Alternatively, it is also possible to use a single LAN interface for both public and private connectivity, but this is not recommended for redundancy reasons.

Figure 1-9 shows what happens when node1 (or target1) fails. Node2 is monitoring node1 and does not hear any heartbeats, so it declares target1 failed (see the right side of Figure 1-9). At this point, the client traffic goes to node2 (target2).

Figure 1-9 Public and Private Interface and a Failover



Heartbeats

From a network design point of view, the type of heartbeats used by the application often decide whether the connectivity between the servers can be routed. For *local* clusters, it is almost always assumed that the two or more servers communicate over a Layer 2 link, which can be either a direct cable or simply a VLAN.

The following traffic traces provide a better understanding of the traffic flows between the nodes:

```
1.1.1.11 1.1.1.10    UDP      Source port: 3343  Destination port: 3343
1.1.1.10 1.1.1.11    UDP      Source port: 3343  Destination port: 3343
1.1.1.11 1.1.1.10    UDP      Source port: 3343  Destination port: 3343
1.1.1.10 1.1.1.11    UDP      Source port: 3343  Destination port: 3343
```

1.1.1.10 and 1.1.1.11 are the IP addresses of the servers on the private network. This traffic is unicast. If the number of servers is greater or equal to three, the heartbeat mechanism typically changes to multicast. The following is an example of how the server-to-server traffic might appear on either the public or the private segment:

```
11.20.40.5  239.255.240.185  UDP  Source port: 3343  Destination port: 3343
11.20.40.6  239.255.240.185  UDP  Source port: 3343  Destination port: 3343
11.20.40.7  239.255.240.185  UDP  Source port: 3343  Destination port: 3343
```

The 239.255.x.x range is the site local scope. A closer look at the payload of these UDP frames reveals that the packet has a time-to-live (TTL)=1:

```
Internet Protocol, Src Addr: 11.20.40.5 (11.20.40.5), Dst Addr: 239.255.240.185
(239.255.240.185)
[…]
Fragment offset: 0
Time to live: 1
Protocol: UDP (0x11)
Source: 11.20.40.5 (11.20.40.5)
Destination: 239.255.240.185 (239.255.240.185)
```

The following is another possible heartbeat that you may find:

```
11.20.40.5  224.0.0.127  UDP Source port: 23  Destination port: 23
11.20.40.5  224.0.0.127  UDP Source port: 23  Destination port: 23
11.20.40.5  224.0.0.127  UDP Source port: 23  Destination port: 23
```

The 224.0.0.127 address belongs to the link local address range, which is generated with TTL=1.

These traces show that the private network connectivity between nodes in a cluster typically requires Layer 2 adjacency between the nodes; in other words, a non-routed VLAN. The Design chapter outlines options where routing can be introduced between the nodes when certain conditions are met.

Layer 2 or Layer 3 Connectivity

Based on what has been discussed in Virtual IP Address, page 1-9 and Heartbeats, page 1-11, you can see why Layer 2 adjacency is required between the nodes of a local cluster. The documentation from the cluster software vendors reinforces this concept.

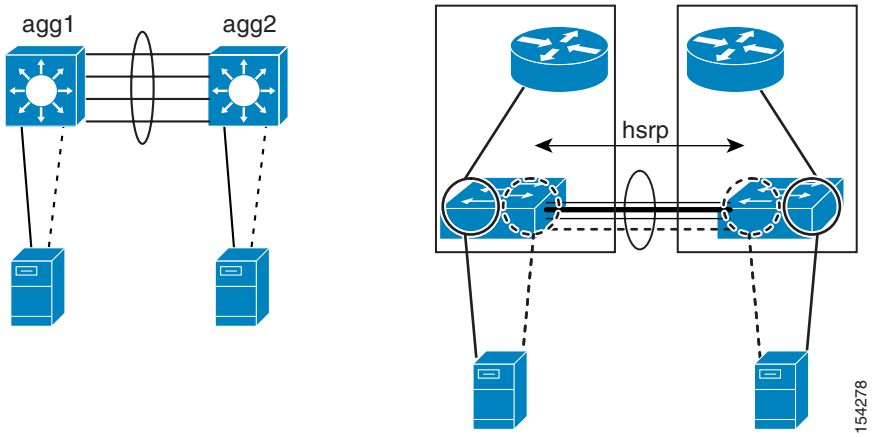
Quoting from the IBM HACMP documentation: “Between cluster nodes, do not place intelligent switches, routers, or other network equipment that do not transparently pass through UDP broadcasts and other packets to all cluster nodes. This prohibition includes equipment that optimizes protocol such as Proxy ARP and MAC address caching, transforming multicast and broadcast protocol requests into unicast requests, and ICMP optimizations.”

Quoting from the MSCS documentation: “The private and public network connections between cluster nodes must appear as a single, non-routed LAN that uses technologies such as virtual LANs (VLANs). In these cases, the connections network must be able to provide a guaranteed, maximum round-trip latency between nodes of no more than 500 milliseconds. The Cluster Interconnect must appear as a standard LAN”. For more information, see the following URL: http://support.microsoft.com/kb/280743/EN-US/. According to Microsoft, future releases might address this restriction to allow building clusters across multiple L3 hops.

Note Some Cisco technologies can be used in certain cases to introduce Layer 3 hops in between the nodes. An example is a feature called Local Area Mobility (LAM). LAM works for unicast traffic only and it does not necessarily satisfy the requirements of the software vendor because it relies on Proxy ARP.

As a result of this requirement, most cluster networks are currently similar to those shown in Figure 1-10; to the left is the physical topology, to the right the logical topology and VLAN assignment. The continuous line represents the public VLAN, while the dotted line represents the private VLAN segment. This design can be enhanced when using more than one NIC for the private connection. For more details, see Complete Design, page 1-22.

Figure 1-10 Typical LAN Design for HA Clusters



Disk Considerations

Figure 1-7 displays a typical failover of a group. The disk ownership is moved from node1 to node2. This procedure requires that the disk be shared between the two nodes, such that when node2 becomes active, it has access to the same data as node1. Different clusters provide this functionality differently: some clusters follow a *shared disk* architecture where every node can write to every disk (and a sophisticated lock mechanism prevents inconsistencies which could arise from concurrent access to the same data), or *shared nothing*, where only one node owns a given disk at any given time.

Shared Disk

With either architecture (shared disk or shared nothing), from a storage perspective, the disk needs to be connected to the servers in a way that any server in the cluster can access it by means of a simple software operation.

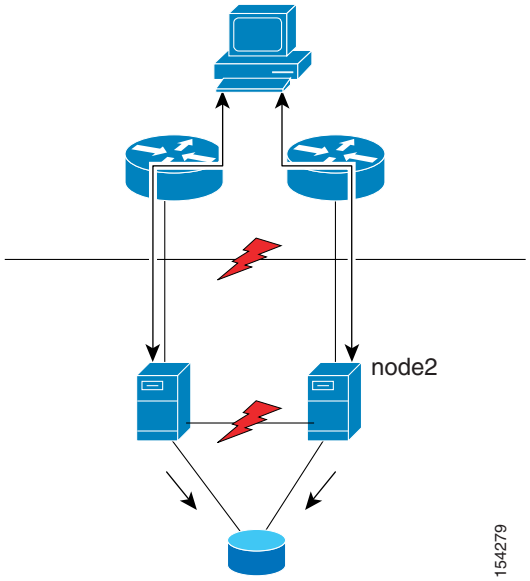
The disks to which the servers connect are typically protected with redundant array of independent disks (RAID): RAID1 at a minimum, or RAID01 or RAID10 for higher levels of I/O. This approach minimizes the chance of losing data when a disk fails as the disk array itself provides disk redundancy and data mirroring.

You can provide access to shared data also with a shared SCSI bus, network access server (NAS), or even with iSCSI.

Quorum Concept

Figure 1-11 shows what happens if all the communication between the nodes in the cluster is lost. Both nodes bring the same group online, which results in an active-active scenario. Incoming requests go to both nodes, which then try to write to the shared disk, thus causing data corruption. This is commonly referred to as the *split-brain* problem.

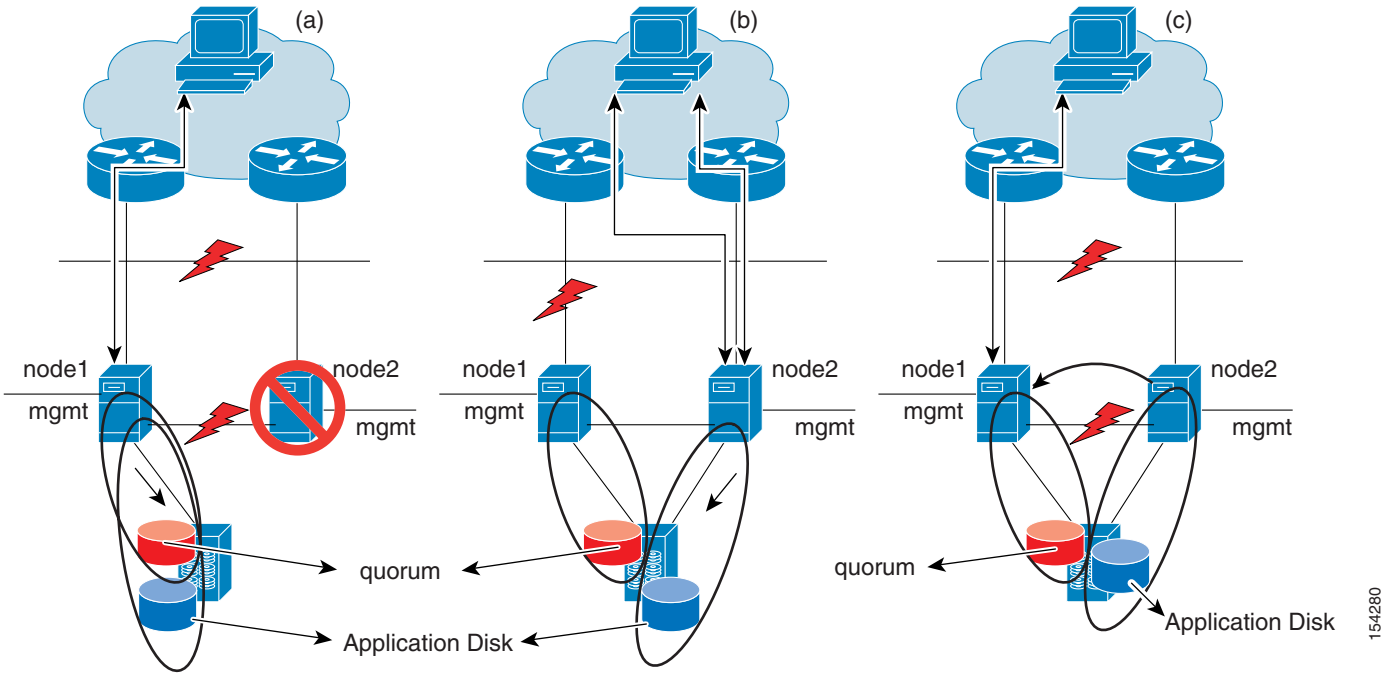
Figure 1-11 Theoretical Split-Brain Scenario



The mechanism that protects against this problem is the *quorum*. For example, MSCS has a *quorum disk* that contains the database with the cluster configuration information and information on all the objects managed by the clusters.

Only one node in the cluster owns the quorum at any given time. Figure 1-12 shows various failure scenarios where despite the fact that the nodes in the cluster are completely isolated, there is no data corruption because of the quorum concept.

Figure 1-12 LAN Failures in Presence of Quorum Disk



In scenario (a), node1 owns the quorum and that is also where the group for the application is active. When the communication between node1 and node2 is cut, nothing happens; node2 tries to reserve the quorum, but it cannot because the quorum is already owned by node1.

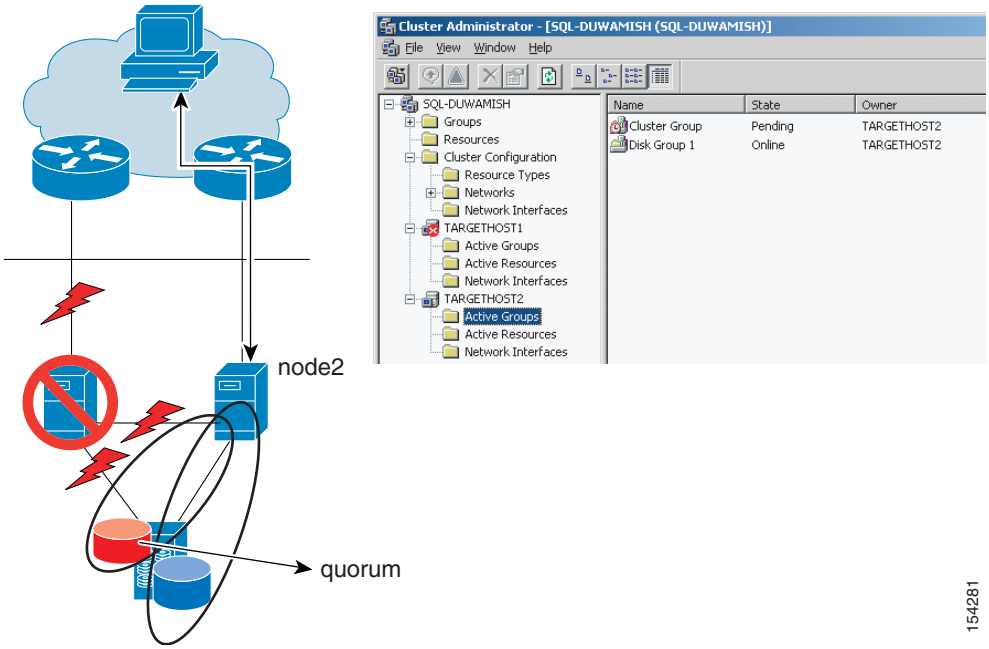
Scenario (b) shows that when node1 loses communication with the public VLAN, which is used by the *application group,* it can still communicate with node2 and instruct node2 to take over the disk for the application group. This is because node2 can still talk to the default gateway. For management purposes, if the quorum disk as part of the *cluster group* is associated with the public interface, the quorum disk can also be transferred to node2, but it is not necessary. At this point, client requests go to node2 and everything works.

Scenario (c) shows what happens when the communication is lost between node1 and node2 where node2 owns the application group. Node1 owns the quorum, thus it can bring resources online, so the application group is brought up on node1.

The key concept is that when all communication is lost, the node that owns the quorum is the one that can bring resources online, while if partial communication still exists, the node that owns the quorum is the one that can initiate the move of an application group.

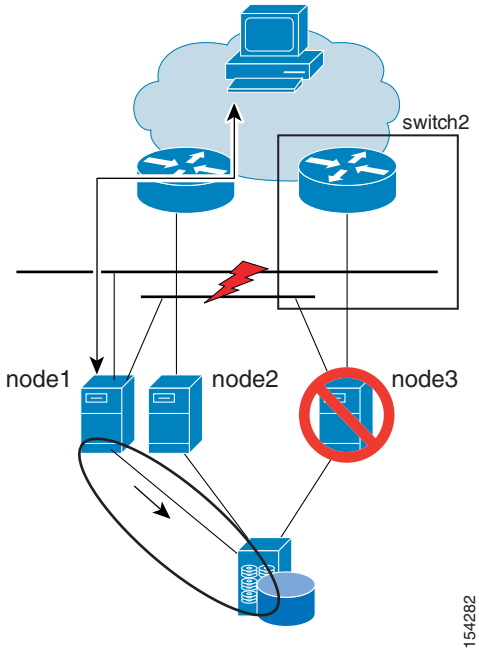
When all communication is lost, the node that does not own the quorum (referred to as the *challenger*) performs a SCSI reset to get ownership of the quorum disk. The owning node (referred to as the *defender*) performs SCSI reservation at the interval of 3s,and the challenger retries after 7s. As a result, if a node owns the quorum, it still holds it after the communication failure. Obviously, if the defender loses connectivity to the disk, the challenger can take over the quorum and bring all the resources online. This is shown in Figure 1-13.

Figure 1-13 Node1 Losing All Connectivity on LAN and SAN



There are several options related to which approach can be taken for the quorum implementation; the quorum disk is just one option. A different approach is the *majority node set*, where a copy of the quorum configuration is saved on the local disk instead of the shared disk. In this case, the arbitration for which node can bring resources online is based on being able to communicate with at least more than half of the nodes that form the cluster. Figure 1-14 shows how the majority node set quorum works.

Figure 1-14 Majority Node Set Quorum



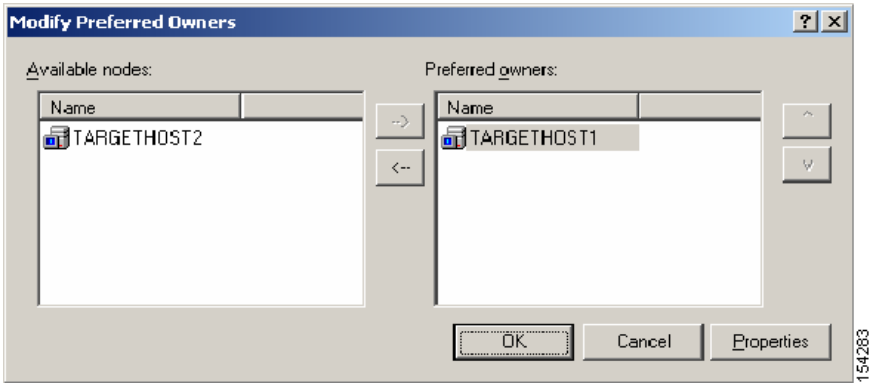
Each local copy of the quorum disk is accessed via the network by means of server message block (SMB). When nodes are disconnected, as in Figure 1-14, node1 needs to have the vote of the majority of the nodes to be the master. This implies that this design requires an odd number of nodes. Also notice that there is no quorum disk configured on the storage array.

Network Design Considerations

Routing and Switching Design

Figure 1-12 through Figure 1-14 show various failure scenarios and how the quorum concept helps prevent data corruption. As the diagrams show, it is very important to consider the implications of the routing configuration, especially when dealing with a geocluster (see subsequent section in this document). It is very important to match the routing configuration to ensure that the traffic enters the network from the router that matches the node that is preferred to own the quorum. By matching quorum and routing configuration, when there is no LAN connectivity, there is no chance that traffic is routed to the node whose resources are offline. Figure 1-15 shows how to configure the preferred owner for a given resource; for example, the quorum disk. This configuration needs to match the routing configuration.

Figure 1-15 Configuring the Preferred Owner for a Resource—Quorum Example

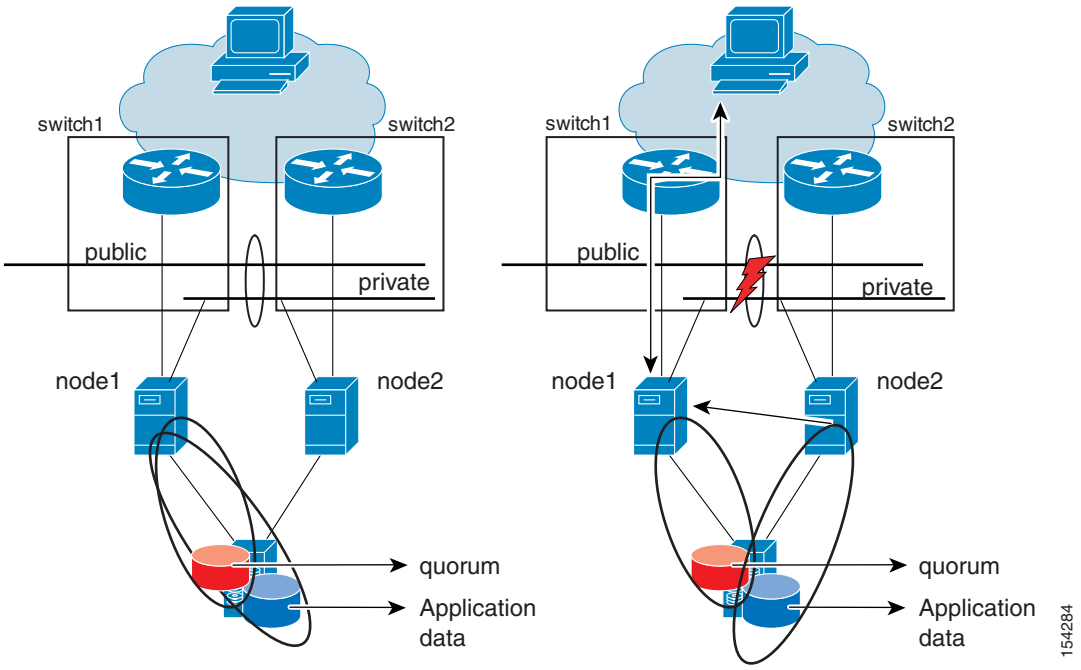


Controlling the inbound traffic from a routing point of view and matching the routing configuration to the quorum requires the following:

- Redistributing the connected subnets
- Filtering out the subnets where there are no clusters configured (this is done with route maps)
- Giving a more interesting cost to the subnets advertised by switch1

Figure 1-16 (a) shows a diagram with the details of how the public and private segment map to a typical topology with Layer 3 switches. The public and private VLANs are trunked on an EtherChannel between switch1 and switch2. With this topology, when the connectivity between switch1 and switch2 is lost, the nodes cannot talk with each other on either segment. This is actually preferable to having a LAN disconnect on only, for example, the public segment. The reason is that by losing both segments at the same time, the topology converges as shown in Figure 1-16 (b) no matter which node owned the disk group for the application.

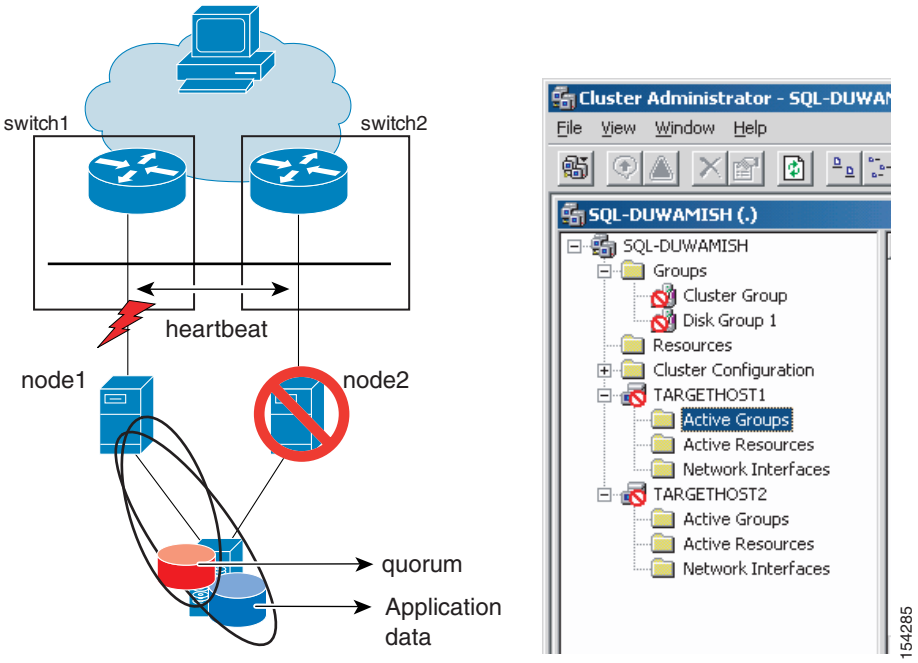
Figure 1-16 Typical Local Cluster Configuration



Importance of the Private Link

Figure 1-17 shows the configuration of a cluster where the public interface is used both for client-to-server connectivity and for the heartbeat/interconnect. This configuration does not protect against the failure of a NIC or of the link that connects node1 to the switch. This is because the node that owns the quorum cannot instruct the other node to take over the application group. The result is that both nodes in the cluster go offline.

Figure 1-17 Cluster Configuration with a Promiscuous Port—No Private Link



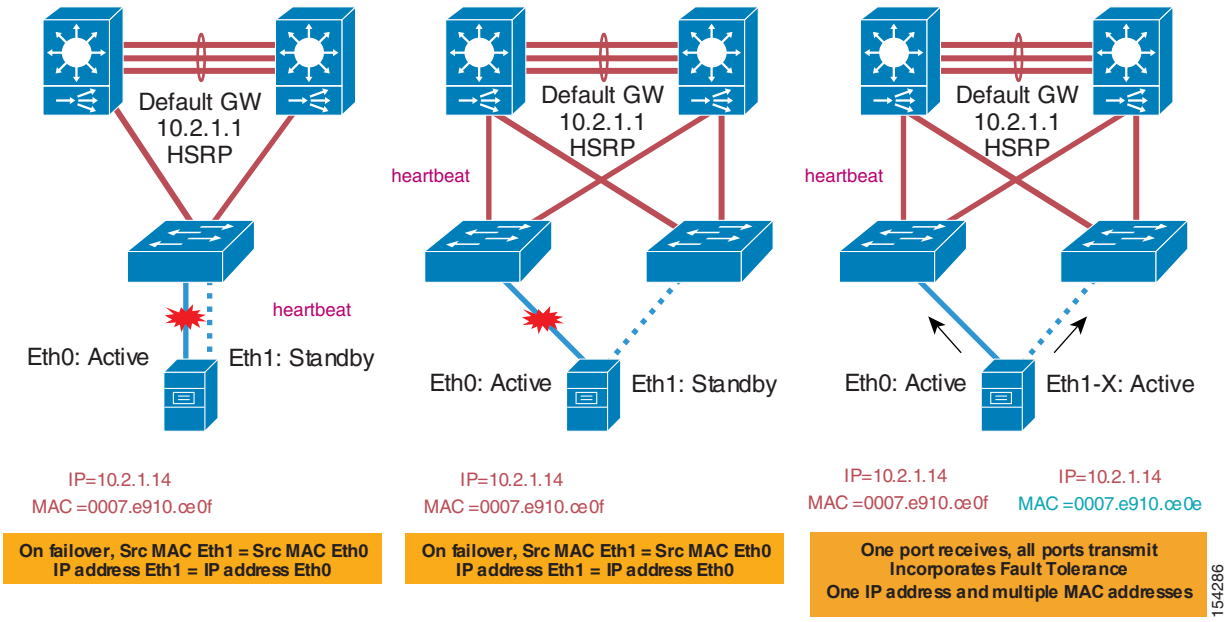
For this reason, Cisco highly recommends using at least two NICs; one for the public network and one for the private network, even if they both connect to the same switch. Otherwise, a single NIC failure can make the cluster completely unavailable, which is exactly the opposite of the purpose of the HA cluster design.

NIC Teaming

Servers with a single NIC interface can have many single points of failure, such as the NIC card, the cable, and the switch to which it connects. NIC teaming is a solution developed by NIC card vendors to eliminate this single point of failure by providing special drivers that allow two NIC cards to be connected to two different access switches or different line cards on the same access switch. If one NIC card fails, the secondary NIC card assumes the IP address of the server and takes over operation without disruption. The various types of NIC teaming solutions include active/standby and active/active. All solutions require the NIC cards to have Layer 2 adjacency with each other.

Figure 1-18 shows examples of NIC teaming configurations.

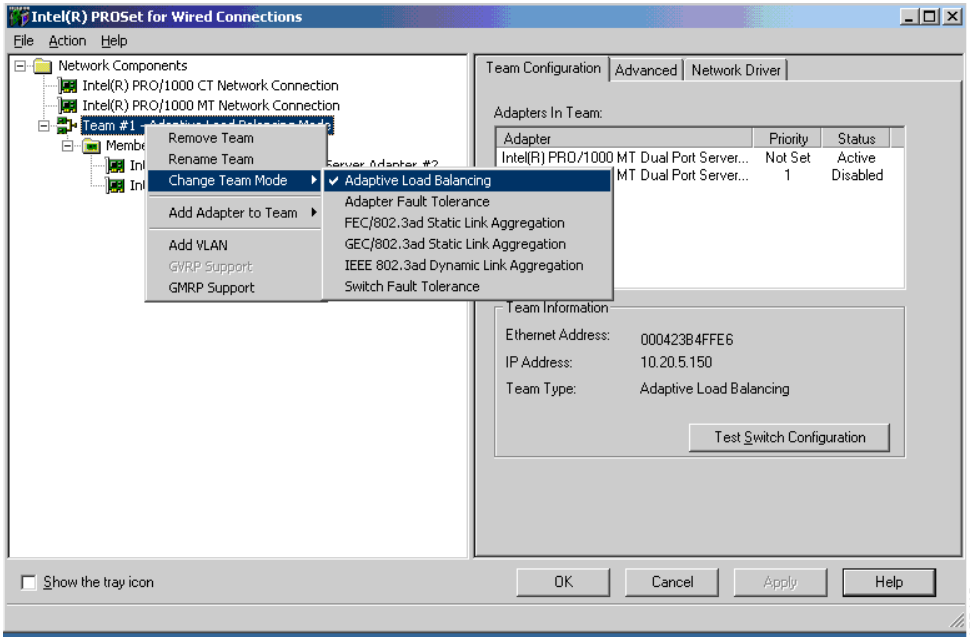
Figure 1-18 NIC Teaming Configurations



With Switch Fault Tolerance (SFT) designs, one port is active and the other is standby, using one common IP address and MAC address. With Adaptive Load Balancing (ALB) designs, one port receives and all ports transmit using one IP address and multiple MAC addresses.

Figure 1-19 shows an Intel NIC teaming software configuration where the user has grouped two interfaces (in this case from the same NIC) and has selected the ALB mode.

Figure 1-19 Typical NIC Teaming Software Configuration



Depending on the cluster server vendor, NIC teaming may or may not be supported. For example, in the case of MSCS, teaming is supported for the public-facing interface but not for the private interconnects. For this reason, it is advised to use multiple links for the private interconnect, as described at the following URL: http://support.microsoft.com/?id=254101.

Quoting from Microsoft: "Microsoft does not recommend that you use any type of fault-tolerant adapter or "Teaming" for the heartbeat. If you require redundancy for your heartbeat connection, use multiple network adapters set to Internal Communication Only and define their network priority in the cluster configuration. Issues have been seen with early multi-ported network adapters, so verify that your firmware and driver are at the most current revision if you use this technology. Contact your network adapter manufacturer for information about compatibility on a server cluster. For more information, see the following article in the Microsoft Knowledge Base: 254101 Network Adapter Teaming and Server Clustering."

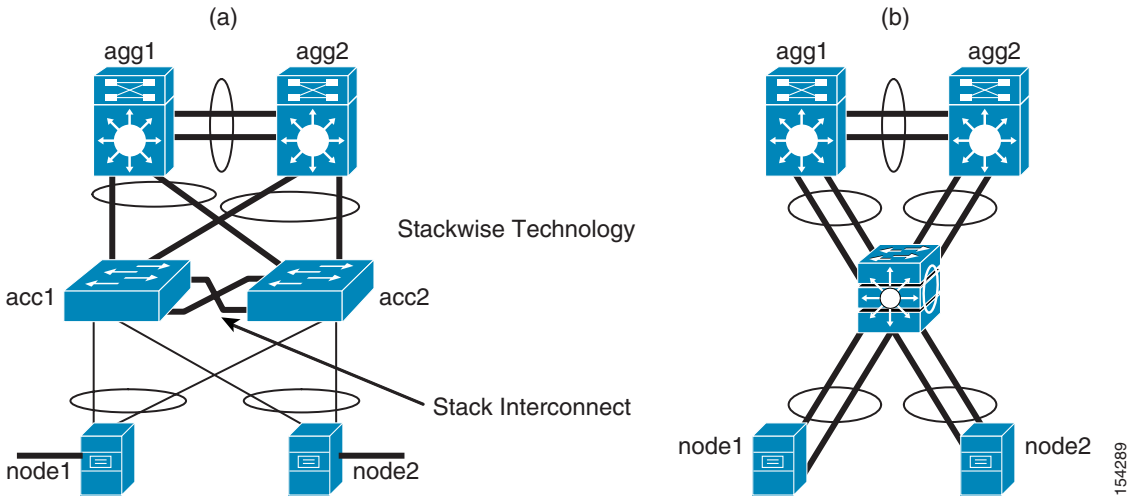
Another variation to the NIC teaming configuration consists in using *cross-stack EtherChannels*. For more information, see the following URL: http://www.cisco.com/en/US/docs/switches/lan/catalyst3750/software/release/12.2_25_sed/configuration/guide/swethchl.html.

Figure 1-20 (a) shows the network design with cross-stack EtherChannels. You need to use two or more Cisco Catalyst 3750 switches interconnected with the appropriate stack interconnect cable, as described at the following URL: http://www.cisco.com/en/US/docs/switches/lan/catalyst3750/software/release/12.2_25_sed/configuration/guide/swstack.html.

The aggregation switches are dual-connected to each stack member (access1 and access2); the servers are similarly dual-connected to each stack member. EtherChanneling is configured on the aggregation switches as well as the switch stack. Link Aggregation Protocol is not supported across switches, so the channel group must be configured in mode "on". This means that the aggregation switches also need to be configured with the channel group in mode on.

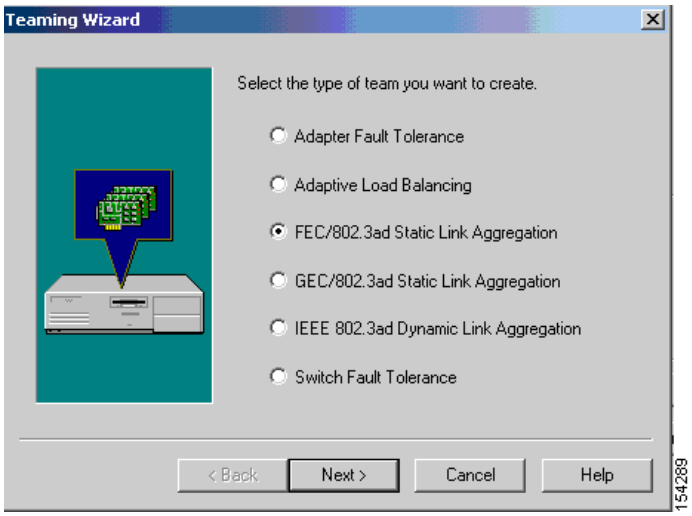
Figure 1-20 (b) shows the resulting equivalent topology to Figure 1-20 (a) where the stack of access switches appears as a single device to the eyes of the aggregation switches and the servers.

Figure 1-20 Configuration with Cross-stack EtherChannels



Configuration of the channeling on the server requires the selection of *Static Link Aggregation*; either FEC or GEC, depending on the type of NIC card installed, as shown in Figure 1-21.

Figure 1-21 Configuration of EtherChanneling on the Server Side



Compared with the ALB mode (or TLB, whichever name the vendor uses for this mechanism), this deployment has the advantage that all the server links are used both in the outbound and inbound direction, thus providing a more effective load balancing of the traffic. In terms of high availability, there is little difference with the ALB mode:

- With the stackwise technology, if one of the switches in the stack fails (for example the master), the remaining one takes over Layer 2 forwarding in 1s (see the following URL: http://www.cisco.com/en/US/docs/switches/lan/catalyst3750/software/release/12.2_25_sed/configuration/guide/swintro.html)

  The FEC or GEC configuration of the NIC teaming driver stops using the link connecting to the failed switch and continues on the remaining link.

- With an ALB configuration, when access1 fails, the teaming software simply forwards the traffic on the remaining link.

In both cases, the traffic drop amounts to few seconds.

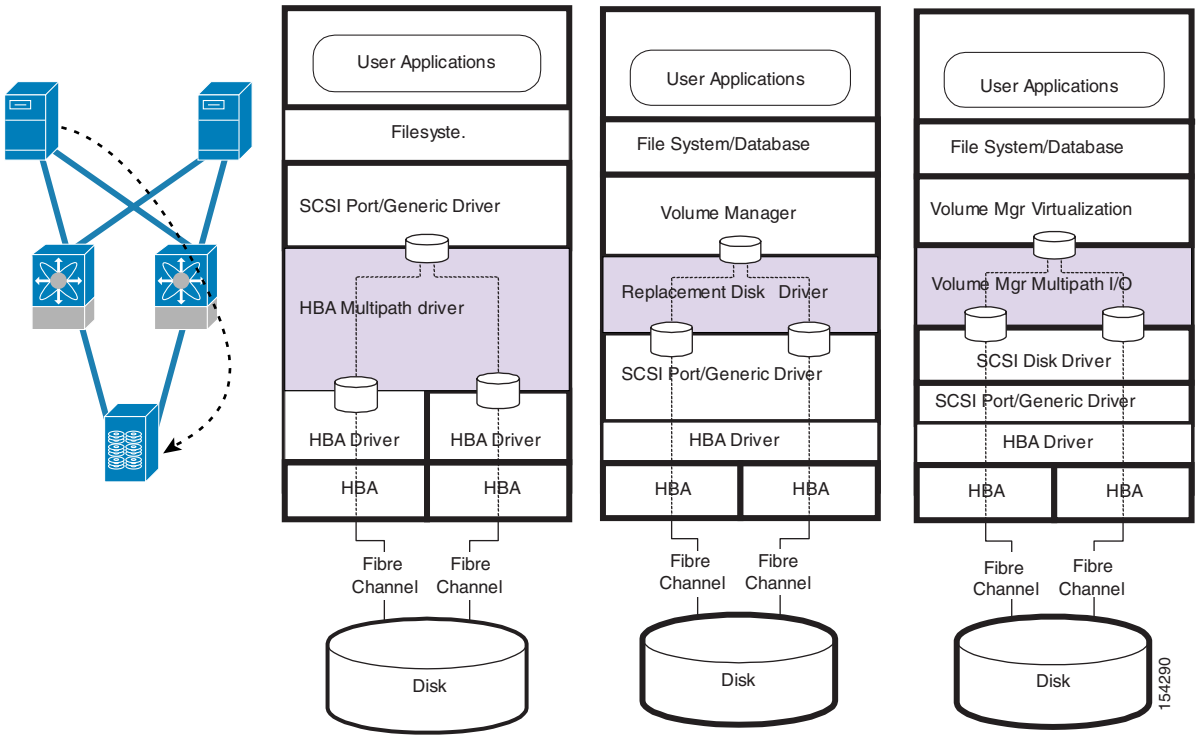
Storage Area Network Design

From a SAN point of view, the key requirement for HA clusters is that both nodes need to be able to see the same storage. Arbitration of which node is allowed to write to the disk happens at the cluster software level, as previously described in Quorum Concept, page 1-13.

HA clusters are often configured for multi-path I/O (MPIO) for additional redundancy. This means that each server is configured with two host-based adapters (HBAs) and connects to two fabrics. The disk array is in turn connected to each fabric. This means that each server has two paths to the same LUN. Unless special MPIO software is installed on the server, the server thinks that each HBA gives access to a different disk.

The MPIO software provides a single view of the disk via these two paths and load balancing between them. Two examples of this type of software include EMC Powerpath and HP Autopath. The MPIO software can be provided by HBA vendors, storage vendors, or by Volume Manager vendors. Each product operates in a different layer of the stack, as shown in Figure 1-22. Several mechanisms can be used by this software to identify the same disk that appears on two different HBAs.

Figure 1-22 SAN Configuration



MPIO can use several load distribution/HA algorithms: Active/Standby, Round Robin, Least I/O (referred to the path with fewer I/O requests), or Least Blocks (referred to the path with fewer blocks).

Not all MPIO software is compatible with clusters, because sometimes the locking mechanisms required by the cluster software cannot be supported with MPIO. To discover whether a certain cluster software is compatible with a specific MPIO solution, see the hardware and software compatibility matrix provided by the cluster vendor. As an example, in the case of Microsoft, see the following URLs:

- http://www.microsoft.com/whdc/hcl/search.mspx
- http://www.microsoft.com/windows2000/datacenter/HCL/default.asp
- http://www.microsoft.com/WindowsServer2003/technologies/storage/mpio/faq.mspx

Besides verifying the MPIO compatibility with the cluster software, it is also important to verify which mode of operation is compatible with the cluster. For example, it may be more likely that the active/standby configuration be compatible than the load balancing configurations.

Besides MPIO, the SAN configuration for cluster operations is fairly simple; you just need to configure zoning correctly so that all nodes in the cluster can see the same LUNs, and similarly on the storage array, LUN masking needs to present the LUNs to all the nodes in the cluster (if MPIO is present, the LUN needs to mapped to each port connecting to the SAN).

Complete Design

Figure 1-23 shows the end-to-end design with a typical data center network. Each clustered server is dual-homed to the LAN and to the SAN. NIC teaming is configured for the public interface; with this design, it might be using the ALB mode (also called TLB depending on the NIC vendor) to take

advantage of the forwarding uplinks of each access switch; MPIO is configured for storage access. The private connection is carried on a different port. If you require redundancy for the private connection, you would configure an additional one, without the two being teamed together.

Figure 1-23 Design Options with Looped Access with (b) being the Preferred Design

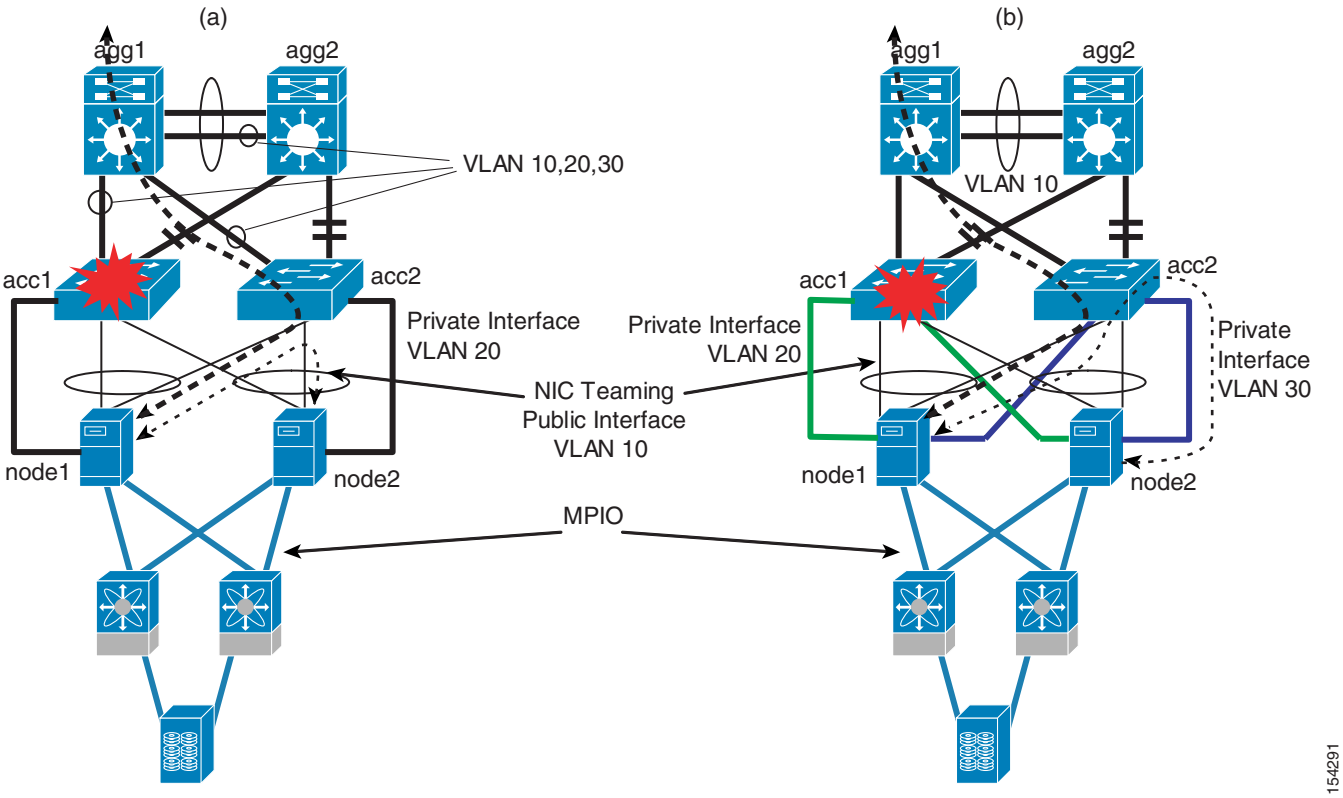
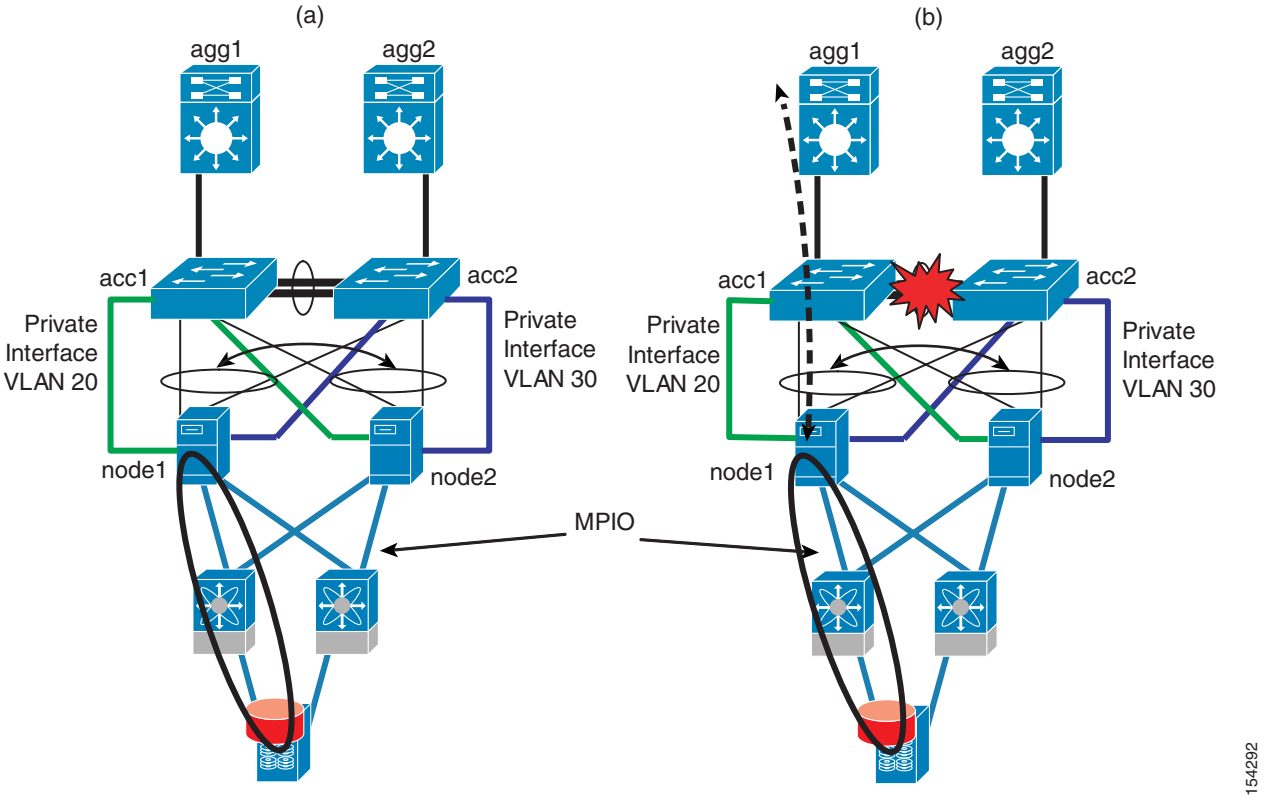


Figure 1-23 (a) shows a possible design where each server has a private connection to a single switch. This design works fine except when one of the two switches fails, as shown. In this case, the heartbeat (represented as the dash line in the picture) needs to traverse the remaining link in the teamed public interface. Depending on the clustering software vendor, this configuration might or might not work. As previously stated, Microsoft, for example, does not recommend carrying the heartbeat on a teamed interface. Figure 1-23 (b) shows a possible alternative design with redundancy on the private links. In this case, there are three VLANs: Vlan 10 for the public interface, and VLAN 20 and 30 for the private links. VLAN 20 is local to the access switch to the left and VLAN 30 is local to the access switch to the right. Each node has a private link to each access switch. In case one access switch fails, the heartbeat communication (represented as the dash line in the picture) continues on the private links connected to the remaining access switch.

Figure 1-24 (a) shows the design with a loop-free access.

Figure 1-24 Design with a Loop-free Access (a) and an Important Failure Scenario (b)



This design follows the same strategy as Figure 1-23 (b) for the private links. The teaming configuration most likely leverages Switch Fault Tolerance, because there is no direct link between the access switch to the right towards the left aggregation switch where HSRP is likely to be the primary. One important failure scenario is the one shown in Figure 1-24 (b) where the two access switches are disconnected, thus creating a split subnet. To address this problem and make sure that the cluster can continue to work, it may be a good design best practice to match the preferred owner for the quorum disk to the aggregation switch that advertises the path with the best metric. This configuration is not the normal default configuration for the aggregation switches/routers. You have to explicitly configure the routing in a way that aggregation1 is the preferred path to the cluster. This is achieved, for example, by using the command **redistribute connected** to filter out all the subnets except the cluster subnet, and by using route maps to assign a better cost to the route advertised by agg1 compared to the one advertised by agg2.