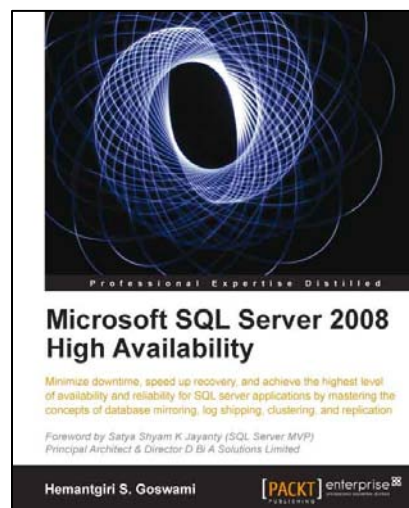


Microsoft SQL Server 2008 High Availability

Hemantgiri S. Goswami



Chapter No.1

"Understanding Windows Domains, Domain Users, and SQL Server Security"

In this package, you will find:

A Biography of the author of the book

A preview chapter from the book, Chapter NO.1 "Understanding Windows Domains, Domain Users, and SQL Server Security"

A synopsis of the book's content

Information on where to buy this book

About the Author

Hemantgiri S. Goswami is an SQL Server MVP, working as a Database Architect in NetDScure Services, Surat, India. He has been a Microsoft SQL Server MVP for three years. He moderates multiple SQL Server community forums, including

<http://www.sql-server-performance.com> and <http://www.sqlserver-qa.net>. He actively participates in and speaks at local user group events, organized under the aegis of <http://www.surat-user-group.org>, the Surat chapter of DotNetChaps and CSI, of which he is a founding and active member. He regularly publishes articles on his blog space <http://www.sql-server-citation.com>. He has recently taken up a new initiative—blogging about SQL in his mother tongue Gujarati, through his blog at <http://sqlservercitation-gujarati.blogspot.com>.

For More Information:

www.PacktPub.com/microsoft-sql-server-2008-high-availabilitybook

He has more than 11 years of experience in the IT industry, for nine years of which he was working as a hardcore DBA focusing on High Availability area. During his stint of 11 years in the IT field, he has worked with the top five IT giants of India. In addition to SQL Server, Hemantgiri also possesses excellent knowledge of Windows Server OS(s) and Networking.

In his free time, he prefers watching cartoons, reading, and even sketching at times. You can reach him via his blog, on Twitter, or by writing to him at hemantgiri@sql-server-citation.com.

For More Information:

www.PacktPub.com/microsoft-sql-server-2008-high-availabilitybook

Microsoft SQL Server 2008

High Availability

The term **High Availability** means that the servers or systems that host or run the business-critical applications should be highly available 24 X 7. As the word itself defines how important it is to make these applications and data available for end-users as well as business users, if this data is not available for a short time, it will be a big problem for both sets of users. Imagine a bank spread across the country and having a huge customer base. One fine day, their server crashes! If the bank relies only on backups, then it might end up losing approximately 15 to 30 minutes of data, depending on the backup strategy. However, the HA options related to SQL Server such as clustering, replication, log shipping, and database mirroring will help overcome this situation.

By the end of the book, you will be able to find yourself in a position where you can easily install and configure the different High Available solutions for SQL Server. You will also be able to troubleshoot most common issues yourself by following the troubleshooting appendix.

What This Book Covers

Chapter 1, Understanding Windows Domains, Domain Users, and SQL Server Security, will help you understand what is Windows domain, what are domain users, and the basic security concepts for Windows and SQL Server to get yourself prepared for the next chapter.

Chapter 2, Implementing Clustering, will help you understand the prerequisites for SQL Server Clustering and guide you on how to install and configure SQL Server Cluster using both T-SQL and SSMS. The chapter also helps you on how to add or remove a node from an existing cluster.

Chapter 3, Snapshot Replication, will help you understand prerequisites for installing Snapshot Replication using SQL Server. It guides you in installing and configuring Snapshot Replication using both T-SQL and SSMS.

Chapter 4, Transactional Replication, will give you information on how to install and configure Transactional Replication. It also helps you understand how replication works and the different options available to configure and install Transactional Replication.

Chapter 5, Merge Replication, helps you install and configure Merge Replication. It also makes you understand the different components of Merge Replication, and how it works. It guides you on how to configure Merge Replication, using both T-SQL and SSMS.

For More Information:

www.PacktPub.com/microsoft-sql-server-2008-high-availabilitybook

Chapter 6, Peer-to-Peer Replication, explains how to install and configure Peer-to-Peer Replication, using both T-SQL and GUI. It also explains how to add or remove nodes.

Chapter 7, Log Shipping, describes what Log Shipping is, how it works, and what are the prerequisite components for its installation. The chapter also helps understand how to install Log Shipping using both T-SQL and SSMS.

Chapter 8, Database Mirroring, explains what Database Mirroring is all about, how it works, and what are the different components we need to implement it. We also learn different types of Database Mirroring and how to install and configure it using both T-SQL and SSMS.

Appendix A, Troubleshooting, contains the troubleshooting tips for the common issues faced in all of the previous chapters.

Appendix B, External References, contains the external references that we might need to refer, in order to gain further information on topics covered in all of the previous eight chapters.

For More Information:

www.PacktPub.com/microsoft-sql-server-2008-high-availabilitybook

1

Understanding Windows Domains, Domain Users, and SQL Server Security

In this chapter, you will get an introduction to Windows domains, domain users, and SQL Server security. This will make clear and enable you to understand how the SQL Server Security mechanism works and how tightly it is integrated with the Windows domain.

In this chapter, we will learn about most important terms of Windows Servers and SQL Server, which will help us understand clustering in Windows Server as well as SQL server. We will learn about:

- What a Windows domain is and what domain users are
- Various authentication modes in Windows Server
- Authentication modes in SQL Server
- Fixed server and fixed database roles in SQL Server
- What clustering is
- What is new in SQL Server 2008
- How clustering works
- Different types of clustering in SQL Server
- Types of Quorum
- Public and private networks

For More Information:

www.PacktPub.com/microsoft-sql-server-2008-high-availabilitybook

Windows domains and domain users

In the early era of Windows, operating system user were created standalone until Windows NT operating system hit the market. **Windows NT**, that is, **Windows New Technology** introduced some great feature to the world – including domains.

A **domain** is a group of computers that run on Windows operating systems. Amongst them is a computer that holds all the information related to user authentication and user database and is called the **domain controller (server)**, whereas every user who is part of this user database on the domain controller is called a **domain user**. Domain users have access to any resource across the domain and its subdomains with the privilege they have, unlike the standalone user who has access to the resources available to a specific system.

With the release of Windows Server 2000, Microsoft released Active Directory (AD), which is now widely used with Windows operating system networks to store, authenticate, and control users who are part of the domain. A Windows domain uses various modes to authenticate users – encrypted passwords, various handshake methods such as PKI, Kerberos, EAP, SSL certificates, NAP, LDAP, and IP Sec policy – and makes it robust authentication. One can choose the authentication method that suits business needs and based on the environment.

Let's now see various authentication methods in detail.

- **Public Key Infrastructure (PKI):** This is the most common method used to transmit data over insecure channels such as the Internet using digital certificates. It has generally two parts – the public and private keys. These keys are generated by a Certificate Authority, such as, Thawte. Public keys are stored in a directory where they are accessible by all parties. The public key is used by the message sender to send encrypted messages, which then can be decrypted using the private key.
- **Kerberos:** This is an authentication method used in client server architecture to authorize the client to use service(s) on a server in a network. In this method, when a client sends a request to use a service to a server, a request goes to the authentication server, which will generate a session key and a random value based on the username. This session key and a random value are then passed to the server, which grants or rejects the request. These sessions are for certain time period, which means for that particular amount of time the client can use the service without having to re-authenticate itself.
- **Extensible Authentication Protocol (EAP):** This is an authentication protocol generally used in wireless and point-to-point connections.

- **SSL Certificates:** A Secure Socket Layer certificate (SSL) is a digital certificate that is used to identify a website or server that provides a service to clients and sends the data in an encrypted form. SSL certificates are typically used by websites such as GMAIL. When we type a URL and press *Enter*, the web browser sends a request to the web server to identify itself. The web server then sends a copy of its SSL certificate, which is checked by the browser. If the browser trusts the certificate (this is generally done on the basis of the CA and Registration Authority and directory verification), it will send a message back to the server and in reply the web server sends an acknowledgement to the browser to start an encrypted session.
- **Network Access Protection (NAP):** This is a new platform introduced by Microsoft with the release of Windows Server 2008. It will provide access to the client, based on the identity of the client, the group it belongs to, and the level compliance it has with the policy defined by the Network Administrators. If the client doesn't have a required compliance level, NAP has mechanisms to bring the client to the compliance level dynamically and allow it to access the network.
- **Lightweight Directory Access Protocol (LDAP):** This is a protocol that runs over TCP/IP directly. It is a set of objects, that is, organizational units, printers, groups, and so on. When the client sends a request for a service, it queries the LDAP server to search for availability of information, and based on that information and level of access, it will provide access to the client.
- **IP Security (IPSEC):** IP Security is a set of protocols that provides security at the network layer. IP Sec provides two choices:
 - **Authentication Header:** Here it encapsulates the authentication of the sender in a header of the network packet.
 - **Encapsulating Security Payload:** Here it supports encryption of both the header and data.

Now that we know basic information on Windows domains, domain users, and various authentication methods used with Windows servers, I will walk you through some of the basic and preliminary stuff about SQL Server security!

Understanding SQL Server Security

Security!! Now-a-days we store various kinds of information into databases and we just want to be sure that they are secured. Security is the most important word to the IT administrator and vital for everybody who has stored their information in a database as he/she needs to make sure that not a single piece of data should be made available to someone who shouldn't have access. Because all the information stored in the databases is vital, everyone wants to prevent unauthorized access to highly confidential data and here is how security implementation in SQL Server comes into the picture.

With the release of SQL Server 2000, Microsoft (MS) has introduced some great security features such as authentication roles (fixed server roles and fixed database roles), application roles, various permissions levels, forcing protocol encryption, and so on, which are widely used by administrators to tighten SQL Server security.

Basically, SQL Server security has two paradigms: one is SQL Server's own set of security measures and other is to integrate them with the domain. SQL Server has two methods of authentication.

Windows authentication

In Windows authentication mode, we can integrate domain accounts to authenticate users, and based on the group they are members of and level of access they have, DBAs can provide them access on the particular SQL server box.

Whenever a user tries to access the SQL Server, his/her account is validated by the domain controller first, and then based on the permission it has, the domain controller allows or rejects the request; here it won't require separate login ID and password to authenticate. Once the user is authenticated, SQL server will allow access to the user based on the permission it has. These permissions are in form of Roles including Server, fixed DB Roles, and Application roles.

- **Fixed Server Roles:** These are security principals that have server-wide scope. Basically, fixed server roles are expected to manage the permissions at server level. We can add SQL logins, domain accounts, and domain groups to these roles. There are different roles that we can assign to a login, domain account, or group – the following table lists them.

Role name	Permission user can have
Sysadmin	Can perform any activity in the server
Serveradmin	Can change server-wide configuration and shut down server
Securityadmin	Can manage logins and their properties
Processadmin	Can end the process that are running in SQL Server
Setupadmin	Can add or remove linked servers
Bulkadmin	Can run the bulk insert statement
Diskadmin	Can manage the disk files
Dbcreator	Can create, alter, drop, or restore any database
public	Default role assigned to each login

- **Fixed DB Roles:** These are the roles that are assigned to a particular login for the particular database; its scope is limited to the database it has permission to. There are various fixed database roles, including the ones shown in the following table:

Role name	Permission user has
db_accessadmin	Alter any users, create schema, connect
db_backupoperator	Back up database, log, and create checkpoint
db_datareader	Can execute select statement
db_datawriter	Can execute delete, insert, and update statements
db_ddladmin	alter – assembly, asymmetric key, certificate, database DDL trigger, database event, notification, dataspace, fulltext catalog, message type, remote server binding, route, schema, service, symmetric key, checkpoint; create – aggregate, default, function, procedure, queue, rule, synonym, table, table, view, XML schema collection and references
db_denydatareader	Cannot execute select
db_denydatawriter	The role is to revoke the right/permission for select statement
db_owner	Can perform any action in the database
db_securityadmin	Can alter – application role, any role, create schema, view definition
dbm_monitor	Can view most recent status in database mirroring monitor

- **Application Role:** The Application role is a database principal that is widely used to assign user permissions for an application. For example, in a home-grown ERP, some users require only to view the data; we can create a role and add a `db_datareader` permission to it and then can add all those users who require read-only permission.
- **Mixed authentication:** In the Mixed authentication mode, logins can be authenticated by the Windows domain controller or by SQL Server itself. DBAs can create logins with passwords in SQL Server. With the release of SQL Server 2005, MS has introduced `password` policies for SQL Server logins. Mixed mode authentication is used when one has to run a legacy application and it is not on the domain network.

In my opinion, Windows authentication is good because we can use various handshake methods such as PKI, Kerberos, EAP, SSL NAP, LDAP, or IPSEC to tighten the security.

SQL Server 2005 has enhancements in its security mechanisms. The most important features amongst them are password policy, native encryption, separation of users and schema, and no need to provide system administrator (SA) rights to run profiler.

These are good things because SA is the super user, and with the power this account has, a user can do anything on the SQL box, including:

- The user can grant ALTER TRACE permission to users who require to run profiler
- The user can create login and users
- The user can grant or revoke permission to other users

A schema is an object container that is owned by a user and is transferable to any other users. In earlier versions, objects are owned by users, so if the user leaves the company we cannot delete his/her account from the SQL box just because there is some object he/she has created. We first have to change the owner of that object and then we can delete that account. On the other hand, in the case of a schema, we could have dropped the user account because the object is owned by the schema.

Now, SQL Server 2008 will give you more control over the configuration of security mechanisms. It allows you to configure metadata access, execution context, and auditing events using DDL triggers – the most powerful feature to audit any DDL event.

If one wishes to know more about what we have seen till now, he/she can go through the following links:

- <http://www.microsoft.com/sqlserver/2008/en/us/Security.aspx>
- <http://www.microsoft.com/sqlserver/2005/en/us/security-features.aspx>
- <http://technet.microsoft.com/en-us/library/cc966507.aspx>

In this section, we understood the basics of domains, domain users, and SQL Server security. We also learned why security is given so much emphasize these days.

In the next section, we will understand the basics of clustering and its components.

What is clustering?

Before starting with SQL Server clustering, let's have a look at clustering in general and Windows clusters.

The word **Cluster** itself is self-descriptive – a bunch or group. When two or more than two computers are connected to each other by means of a network and share some of the common resources to provide redundancy or performance improvement, they are known as a cluster of computers.

Clustering is usually deployed when there is a critical business application running that needs to be available 24 X 7 or in terminology – High Availability. These clusters are known as Failover clusters because the primary goal to set up the cluster is to make services or business processes that are business critical available 24 X 7. MS Windows server Enterprise and Datacenter edition supports failover clustering. This is achieved by having two identical nodes connected to each other by means of private network or commonly used resources. In case of failure of any common resource or services, the first node (**Active**) passes the ownership to another node (**Passive**).

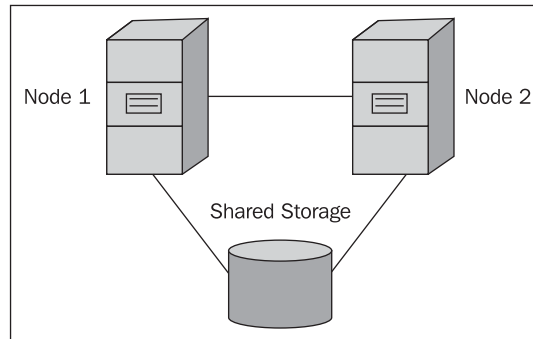
SQL Server Clustering is built on top of Windows Clustering, which means before we go about installing SQL Server clustering, we should have Windows clustering installed. Before we start, let's understand the commonly used shared resources for the cluster server.

Clusters with 2, 4, 8, 12 or 32 nodes can be built Windows Server 2008 R2. Clusters are categorized in the following manner:

- **High-Availability Clusters:**

This type of cluster is known as a Failover cluster. High Availability clusters are implemented when the purpose is to provide highly available services.

For implementing a failover or high availability cluster one may have up to 16 nodes in a Microsoft Cluster. Clustering in Windows operating systems was first introduced with the release of Windows NT 4.0 Enterprise Edition, and was enhanced gradually. Even though we can have non-identical hardware, we should use identical hardware. This is because if the node to which cluster fails over has lower configuration, then we might face degradation in performance.



- **Load Balancing:**

This is the second form of cluster that can be configured. This type of cluster can be configured by linking multiple computers with each other and making use of each resource they need for operation. From the user's point of view, all of these servers/nodes linked to each other are different. However, it is collectively and virtually a single system, with the main goal being to balance work by sharing CPU, disk, and every possible resource among the linked nodes and that is why it is known as a Load Balancing cluster.



SQL Server doesn't support this form of clustering.

- **Compute Clusters:**

When computers are linked together with the purpose of using them for simulation for aircraft, they are known as a compute cluster. A well-known example is Beowulf computers.

- **Grid Computing:**

This is one kind of clustering solution, but it is more often used when there is a dispersed location. This kind of cluster is called a Supercomputer or HPC. The main application is scientific research, academic, mathematical, or weather forecasting where lots of CPUs and disks are required – SETI@home is a well-known example.

If we talk about SQL Server clusters, there are some cool new features that are added in the latest release of SQL Server 2008, although with the limitation that these features are available only if SQL Server 2008 is used with Windows Server 2008. So, let's have a glance at these features:

- **Service SID:** Service SIDs were introduced with Windows Vista and Windows Server 2008. They enable us to bind permissions directly to Windows services. In the earlier version of SQL Server 2005, we need to have a SQL Server Services account that is a member of a domain group so that it can have all the required permissions. This is not the case with SQL Server 2008 and we may choose Service SIDs to bypass the need to provision domain groups.
- **Support for 16 nodes:** We may add up to 16 nodes in our SQL Server 2008 cluster with SQL Server 2008 Enterprise 64-bit edition.
- **New cluster validation:** As a part of the installation steps, a new cluster validation step is implemented. Prior to adding a node into an existing cluster, this validation step checks whether or not the cluster environment is compatible.
- **Mount Drives:** Drive letters are no longer essential. If we have a cluster configured that has limited drive letters available, we may mount a drive and use it in our cluster environment, provided it is associated with the base drive letter.
- **Geographically dispersed cluster node:** This is the super-cool feature introduced with SQL Server 2008, which uses VLAN to establish connectivity with other cluster nodes. It breaks the limitation of having all clustered nodes at a single location.
- **IPv6 Support:** SQL Server 2008 natively supports IPv6, which increases network IP address size to 128 bit from 32 bit.
- **DHCP Support:** Windows server 2008 clustering introduced the use of DHCP-assigned IP addresses by the cluster services and it is supported by SQL Server 2008 clusters. It is recommended to use static IP addresses. This is because if some of our application depends on IP addresses, or in case of failure of renewing IP address from DHCP server, there would be a failure of IP address resources.
- **iSCSI Support:** Windows server 2008 supports iSCSI to be used as storage connection; in earlier versions, only SAN and fibre channels were supported.

How clustering works

A highly available application or system is the key concept in cluster environment. Microsoft SQL Server is a cluster-aware application, and it works well to cater for this business need. Let's see how it works.

Before we go further into details, let's see some common terms here:

- **Active/Passive Cluster:** In this setup, there will be one server that remains idle and takes over the control or the ownership of the resources at the time of failover.
- **Active/Active Cluster:** Here, the only difference is that both the nodes in the cluster are active and running, and the surviving node will take over the control or the ownership of the resources when a failover occurs.
- **Public Network:** This is a network available to external resources or systems.
- **Private Network aka Heartbeat:** This is a network that is available to SQL Server cluster nodes only; heartbeat is used to check the health of another node.
- **Shared Disk Array:** A disk array is nothing but more than one disk used collectively and shared among the cluster nodes. However, at any point of time only one node – the active node or the owner of the resources – can access the disks, in order to protect data from being overwritten.
- **Quorum:** This is the disk resource wherein the status of the cluster is being written, especially by the Windows clustering. Failure of this resource can lead to failure of the entire clustering setup.
- **Cluster Name:** This is the name of a Windows cluster.
- **Cluster IP:** This refers to the IP address on the public network that is used by external systems or client machines to connect to the cluster.
- **Cluster Resource Type:** This can be any resource that can be configured for clustering, that is, a physical disk.
- **Cluster Account:** This is the Administrator account used to control and run the services for a cluster; this account must be configured at the domain level and should be added to the local administrator group in each cluster node.
- **Cluster Group:** This is a kind of container, for example, SQL Server, wherein cluster-aware applications or services are grouped.
- **Cluster Name for Virtual SQL Server:** This is the name of a Virtual SQL Server, which is then used by client machines to connect to.
- **IP for Virtual SQL Server:** This will be the IP address used by SQL Server, and clients use this IP address to connect to SQL Server.
- **Full-text Search:** SQL Server Full-Text search.

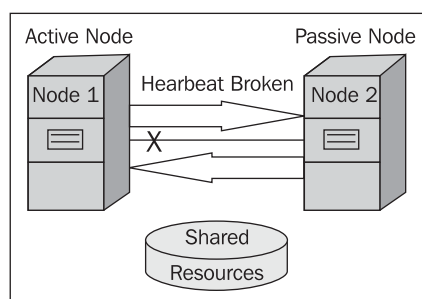
MSDTC

Microsoft Distributed Transaction Coordinator (MSDTC) is a service used by the SQL Server when it is required to have distributed transactions between more than one machine. In a clustered environment, SQL Server service can be hosted on any of the available nodes if the active node fails, and in this case MSDTC comes into the picture in case we have distributed queries and for replication, and hence the MSDTC service should be running.

As we have understood the basics of the components that are used while working with clustering, let's now have a look at how it actually works.

Let's see the example of a Single Node failover cluster. Here, in this case, if anything goes wrong with the active node, the second node will become active and take over the control and ownership of the resources. It is recommended to use fibre channel or SCSI cables for the shared disk arrays for each node. Also, the data should be stored on the shared disk so that it will become accessible by both the nodes in case of failure; however, please note that at any given time only one node can access the disk, in order to protect the data from being overwritten. Apart from these considerations, select a disk system that supports fault tolerance, that is, a RAID array.

So the question arises as to how the passive node senses the failure. Recollect that we just talked about the public and private network (Heartbeat). The public network is exposed to the external resources or computers whereas the private network is shared between cluster nodes. What happens here is, whenever a service or resource gets stressed out or doesn't respond to the private network, that is, its Heartbeat fails, node2 or the passive node initiates the process to take over the ownership of the resources owned by node1. We can refer to the following image:



There are some questions that could be asked. Let's now have a look at some of the main questions:

- **Question:** What will happen to the data that is being accessed?
Answer: Well, this is taken care of by shared disk arrays as it is shared and every node that is part of the cluster can access it; however, one node at a time can access and own it.
- **Question:** What about clients that were connected previously? Does the failover mean that developers will have to modify the connection string?
Answer: Nothing like this happens. SQL Server is installed as a virtual server and it has a virtual IP address and that too is shared by every cluster node. So, the client actually knows only one SQL Server or its IP address. Here are the steps that explain how Failover will work:
 - Node 1 owns the resources as of now, and is active node.
 - The network adapter driver gets corrupted or suffers a physical damage.
 - Heartbeat between Node1 and Node 2 is broken.
 - Node 2 initiates the process to take ownership of the resources owned by the Node 1.
 - It would approximately take two to five minutes to complete the process.

There are mainly two ways that inform whether a failover should occur.

- **Heartbeat:** We have just seen this in the preceding example.
- **Resource-specific detection:** Almost every resource that is part of a cluster has its own specific method such as SQL Server Service, Analysis service, disks, and so on:
 - **SQL Server and Analysis service:** These rely on the network name and IP address. If any of this fails, SQL Server or Analysis service goes offline.
 - **Shared Disks:** There are vendor-provided applications that are cluster-aware and which will check periodically whether or not the resource is available.

Windows server has built-in support called `LooksAlive`, to check every five seconds whether or not the services are running. How it works is, after every five seconds, `SOAGTRES.DLL`, a resource DLL that runs under the cluster account service context, makes a call to the service control manager to check a registry entry to be sure that it is running and sends acknowledgement back to `SOAGTRES.DLL` as either true or false.

`IsAlive` (which occurs every 60 seconds) is a more detailed detection performed by the `SQLSRVRES.DLL` that does the task of verifying that SQL resources are online, registry entries are correct, and SQL Server is running in normal mode. It also checks if the system databases are running normally by executing T-SQL. The `IsAlive` check internally calls Resource Monitor, which reports the status of resources to `SQLSRVRES.DLL`, as either 0 (false) or 1 (true). Resource Monitor depends on the registry for the status information. The status information is compared with the cached value in cluster configuration database; if Resource Monitor returns the status as offline/offline pending, failed, or false, then `SQLSRVRES.DLL` will call the `Online` function to bring the resource back online. If it doesn't succeed in the retries (here it executes `select @@servername`), it finally considers that particular resource as failed and sends it back to Resource Monitor. In turn, this triggers the failover process with the help of the failover manager. If the resource becomes online in the first attempt, or within the retries limit, the failover process doesn't occur; on the other hand, if the resource fails to become online and exceeds the limit of retries, the failover process is initiated.

Types of clusters

There are four types of clustering solutions available with SQL Server 2008, which are the following:

- Failover Cluster: Single-instance Cluster
- Failover Cluster: Multi-instance Cluster
 - All nodes with active instances
 - N+1
- Failover Cluster: Multi-site Cluster
- Failover Cluster: Guest Cluster

Single-instance Cluster

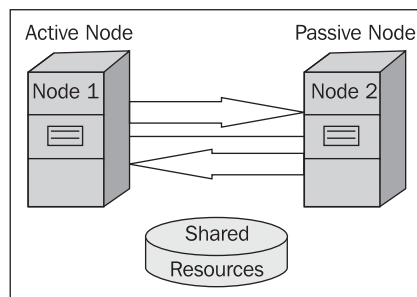
This is the most common and widely used architecture for configuring SQL Server clusters. In this type of clustering, there will be two participating nodes in which one will be active or owner of the resources and the second will be passive. Whenever something crashes or fails, the active node passes ownership of all the resources to the second node—this is called failover.

As there is only a single node that is active or holds a resource like SQL Server services in running mode, this type of clustering is called Single-instance Clustering.



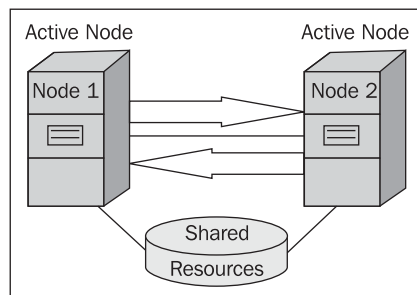
Note: There is a misconception that I often see with the terms **Active/Active** and **Active/Passive**. Let me clarify the terms here, and illustrate why the cluster is being referred to as Active/Passive or Active/Active; however, in any case both are possible with Single-Instance Clusters only.

The most common and widely used configuration is called an Active/Passive cluster (Single-instance Failover Cluster). There will be two nodes that are configured as a cluster node, and one of them is active and the second will remain passive or idle. In case of failure of any kind on Node 1, it fails over to second (or passive) server and now Node 2 will become the primary or active node for the cluster. Refer to the following scenario. If there is a failure in the active node the **Heartbeat** (refer to the *How clustering works* section) will be interrupted and ownership of the any resources will be taken by Node 2, which was passive:



Multi-instance Cluster

In this type of clustering, there can be more than one node failover. This means in case of failure there can be more than one node actively available to take over the ownership of the resources.



As its name says, there will be one or more than one instance or node running in the failover cluster. In the first type of configuration all the nodes are equal. When a node fails, all the failover instances will fail over to another node. What we should remember here is that we need to tolerate multiple node failures. The reason is simple: the node that will take over the ownership should be capable enough of serving the peak hours work load. And, for this, we should use `AntiAffinityClassNames` to set priority.

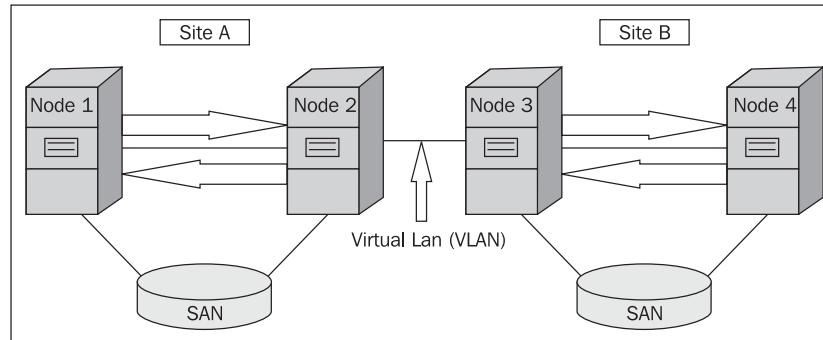
The second type of configuration is referred to as $n+1$. In this type of cluster, there will be a dedicated node available to take on the workload of any other node in case of failure. This configuration increases capacity when all nodes are available, in other words—optimum utilization of resources. We may also have an option to limit the number of dedicated nodes within the total limit of nodes that the SQL Server clustering solution supports. To have better control over instance allocation during failover or balancing, we should use preferred owner or `AntiAffinityClassNames` on the resource group.

In this type of configuration, there will be two nodes that are active and so it can be called a multi-instance cluster. This means that at any given point there will be two separate SQL Server instances running in a clustered environment. In case of failure (say Node 1 fails), all the users, applications, or database(s) that are connected to Node 1 will start using Node 2 and its resources. So, here it is more important to consider that the load might increase on Node 2 while designing or sizing, such as setting minimum and maximum memory; leaving default settings for minimum and maximum memory will consume all available memory and release memory to the OS as and when required, which is not desired. We can refer to the following screenshot.

Multi-site Failover Cluster

This was introduced in SQL Server 2005 clustering, and is also known as **geographically dispersed failover** or **stretch cluster** as it is designed to cope with the situation of a disaster at one site. This is how it functions: There are two sites at physically dispersed locations or sites or datacenters. If there is damage or failure at one site, the other will be up and running, providing a much more robust solution and high availability.

We can refer to the following screenshot:

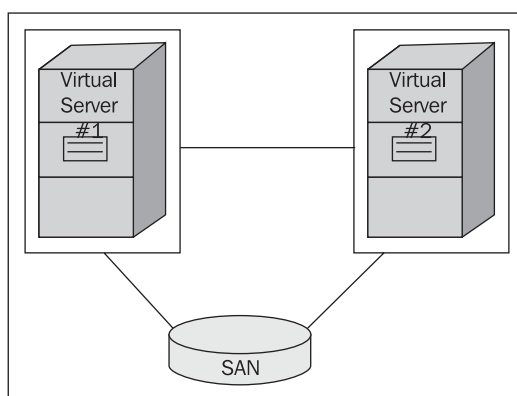


There are six things you should not forget when you decide to use multi-site clustering:

- Our data should get replicated consistently and continuously on the other site. This will ensure that we have the latest data and can tolerate disk failover. If our database and logs are on separate disk drives, we need to ensure that they both get replicated to avoid any data losses and maintain consistency in the state of the data.
- Use Stretch VLAN so that a single subnet masks across all the participating nodes. Windows Server 2008 Cluster supports different subnet masks across cluster nodes; the same feature is yet to be added to SQL Server 2008. Most of the data centers at different locations do not share a single subnet, and it is necessary to have single subnet to have connectivity between these sites.
- Quorum should be configured in a way that either site would run in case of failure of the other. The configuration of the Quorum is the same as it is with single-site failover.
- Heartbeat configuration is supposed to tolerate increased latency; this can be achieved by increasing latency in the Windows server failover cluster inter-node heartbeat so that it does not detect it as stress.
- Ensure that the disk resources are a part of the same group, and there are no non-disk resource dependencies to any of the disks. By default, Node and Disk Majority mode is selected, and in the case of odd number of nodes, Node Majority is selected.
- Having a single subnet will ensure that the store replication and clustering between two geographically dispersed locations is similar to a single-site failover cluster.

Guest Failover Clustering

Guest Failover Clustering is nothing but installing and configuring clustering using virtual machines. In guest failover configuration you may have the cluster nodes on the same physical server, although it is recommended that you configure cluster nodes on different physical servers. The reason is that you will run in to pain when there is damage in the physical box; so to have the application or database available you must configure cluster nodes on separate physical nodes.



Both the versions, SQL Server 2005 and SQL Server 2008, support Guest Failover Clustering. The pre-requisites are:

- The host operating system should be running on a supported virtualization environment such as:
 - Windows Server 2008 with Hyper-V
 - Microsoft Hyper-V server 2008
 - Certified configuration from Server Virtualization Validation Program (SVVP)
- The guest operating system should be running on Windows Server 2008.
- The environment should meet the requirements mentioned in the knowledge-based article at <http://support.microsoft.com/kb/943984>.

Components of clustering

There are some components that we need to carefully consider before we go about designing cluster architecture and installing. Here are the components:

Shared disk array

As the name implies, the disks are shared. The reason behind using shared disk arrays is the fact that they are accessible from remote locations or clients and this is exactly what is required by cluster services.

The cluster architecture forces us to use shared disk resources, as the disk to be used with the cluster environment should be accessible by any of the nodes. The shared disk plays a vital role in the cluster, because it stores not only data files, but logs, FILESTREAM files, and full-text index files. In case of failover, the preferred node will have access and ownership for the disk resource so that clients can access the database system without any interruption.

The shared disk should be reliable and fault tolerant as it stores database files. We can have this shared disk resource in form of Fibre Channel SCSI or SAN.

Internet Small Computer System Interface

In the earlier version of SCSI, we needed to set special cabling, which we need not worry about in the **Internet Small Computer System Interface (iSCSI)** as it works well with IP and that is the beauty of the iSCSI—it is used to transfer data across intranets and to manage storage for long distances. This feature is really very useful and handy in the case of Multi-site clustering because we may use it as a location-independent storage solution.

Also, administrators can consider using iSCSI from the point of view of storage consolidation, that is, dispersed located data center, or multi-site clustering, among others.

Storage Area Network (SAN)

SAN is network storage, which is meant to be used when we need to consolidate our storage at our corporate network or at our data center as a highly reliable storage solution.

SAN has multiple (sometimes hundreds of) hard disks attached to it using high performance adapters. Also, it has a great caching capacity that increases the performance ultimately. All these hard disk drives are then virtualized or created logically thus labeled as **Logical Unit Number (LUN)**. These LUNs can be shared or dedicated to a particular application, such as, SQL Server.

Every operating system maintains its own file system to the dedicated LUNs, that is, NTFS, and when it comes to shared LUNs among different flavors of operating systems, it may use SAN File systems to cater to the need. In most cases, SAN uses fibre channel to connect to the network, called Fibre Channel over Ethernet (FCoE), though it is not limited to FCoE only.

Redundant Array of Independent Disks

Redundant Array of Independent Disks (RAID) provides better fault-tolerance by making use of redundancy of disk(s). RAIDs are widely used as storage solutions to get the best I/O performance, based on the application, whether the application is write intensive or read intensive. We may have a choice having RAID array with iSCSI or SAN. There are mainly three concepts in RAID:

- **Mirroring:** In this type, an entire disk is mirrored on another disk or set of disks using hardware or software-level mirroring. Mirroring creates a replica of what is written on one disk on another disk, or set of disks, and that is why configuring Mirroring requires at least two drives, for example, RAID 1.
- **Striping:** Here, there are two or more disks required to configure striping. While writing data into disks or reading from the disks, it is spread across all the disks configured in stripes and thus gives best performance for writing as each disk has its own reader, for example, RAID 5.
- **Parity:** In general, parity sets status as even or odd by adding a bit known as the parity bit. For even parity, the parity bit is set to 1 if the number of ones in the set of bits (not including the parity bit) is odd. On the other hand for odd parity, the parity bit is set to 1 if the number of ones is even. So, if there is some corruption in data being written or read, it will not match the pattern and thus reports incorrect status, for example, RAID 5. This is generally known as *fault-tolerance*.

There are many RAID arrays available such as RAID 0, RAID 1, RAID 3, RAID 4, RAID 5, RAID 6, RAID 10, and RAID 01. Here, we will discuss only those that are widely used with the SQL Server to boost performance level.

- **RAID1:** This is the simplest form of RAID array. It requires a minimum of two disks to create a RAID1 array. RAID1 creates a replica of the disk being written, and is extremely helpful to recover data in case of physical damage to the first disk. RAID1 is the perfect choice to store operating systems, binaries, and database log files.

- **RAID5:** RAID5, also known as *stripe set with parity*, requires three or more disks, and it creates and stores parity values internally. So in case of failure or damage of a single disk, data can be recalculated and rebuilt on a newly added drive. Please be aware that this will degrade the performance until the disk that has failed or damaged is rebuilt. This type of RAID is most useful when our application is read intensive as it reads data in a sequential manner.
- **RAID10:** RAID10 is also known as *mirrored stripe* or *mirrored then striped*. This means it is a combination of RAID1+RAID0. RAID10 is used when we want best performance with a write-intensive application or data files and tempdb, as it writes data in a random manner.

The Quorum

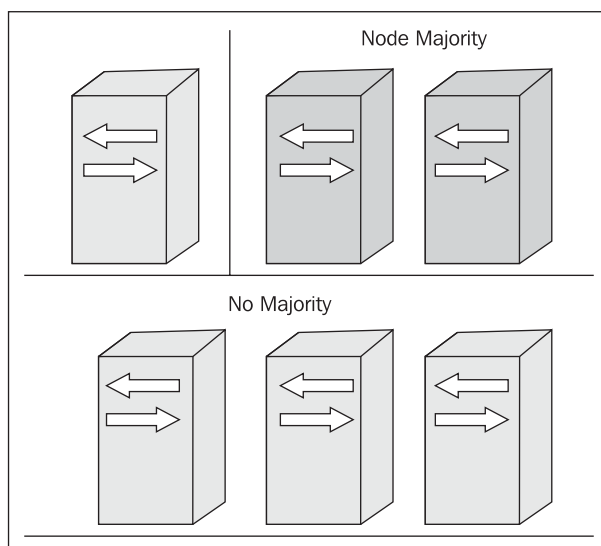
Dictionary says that a **Quorum** is the minimum number that should match to perform a task further or to make a decision. Here, in the case of clustering, the meaning is no different.

As we all know by now, the cluster nodes send and receive information over the network. It is very possible that sometimes a change is done at the node that is the owner or active node, which then fails before sending the data back to another node in the cluster; in this case the cluster will not work. The nodes in clusters vote and based on that it will be decided whether or not the node should continue; for example, there is a four-node cluster and three of them can talk to each other but not the node3.

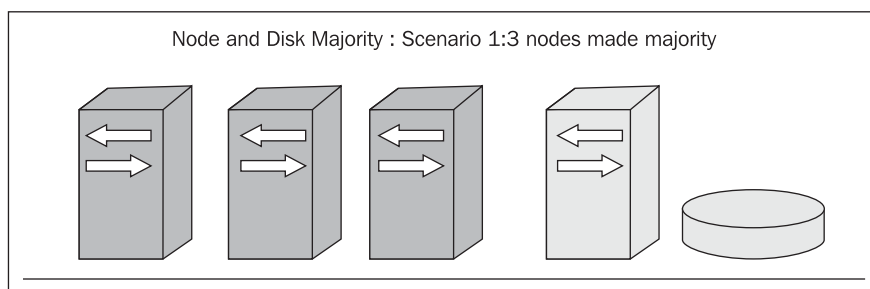
Quorum is a resource where all events pertaining to SQL Server Clusters get logged or recorded on a separate disk. Events can include change in the configuration, in active node, or owner of the resources changed, among others.

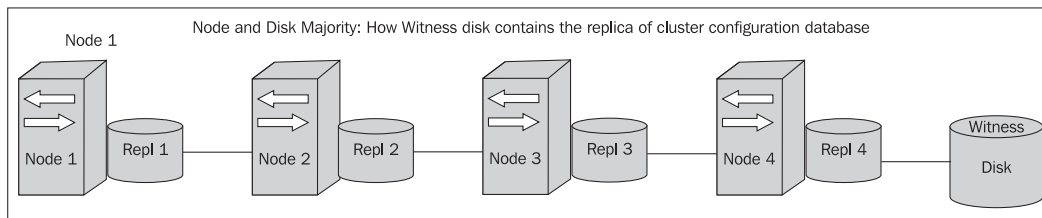
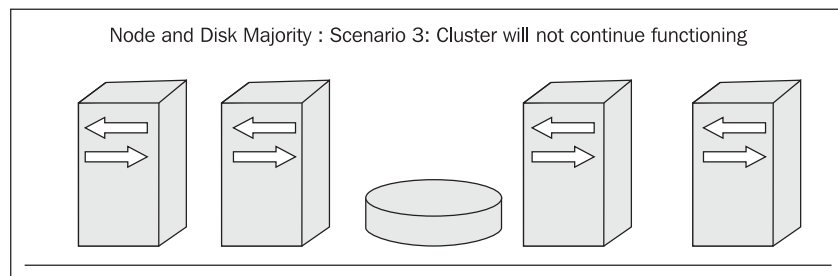
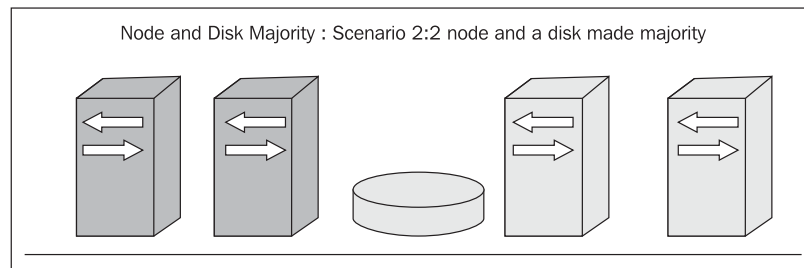
With SQL Server 2008, Microsoft has made significant changes, which will help easily create Quorum on the basis of its application and to do so automatically. There are four types of quorum available, including:

- **Node Majority:** In this mode, each cluster node can vote. Majority is decided upon 50 percent of the nodes. If votes come for 50 percent or greater than 50 percent, then the cluster will function. This is the preferred selection when there is no shared disk or storage provided and numbers of nodes are odd. Consider an example where a three-node cluster is set up. Now if one of the nodes fails to communicate, what will happen? In this case, the votes will be more than 50 percent as two out of three nodes are functioning and communicating with each other and that is why it will continue to function. On the contrary, if only one node is functioning properly of the three, then the cluster will fail to work.



- Node and Disk Majority:** In this configuration, every cluster node plus a designated disk (as a witness) can vote, and here too if more than 50 percent votes are calculated to keep running as a cluster, the cluster will continue to function. This means if there is a four-node cluster with a disk witness, the total number of disks adds up to five as there are four disks on the system plus one witness disk; so, in this case, the cluster will continue to work if three or more votes come.



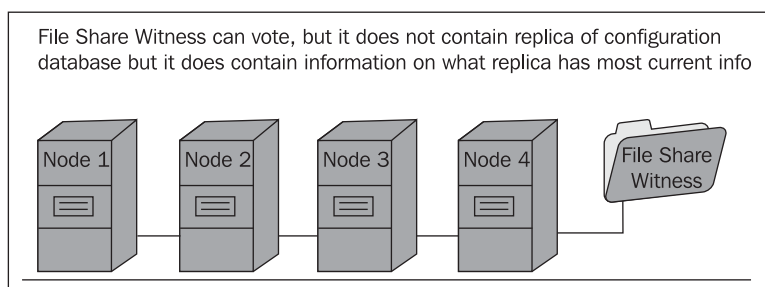
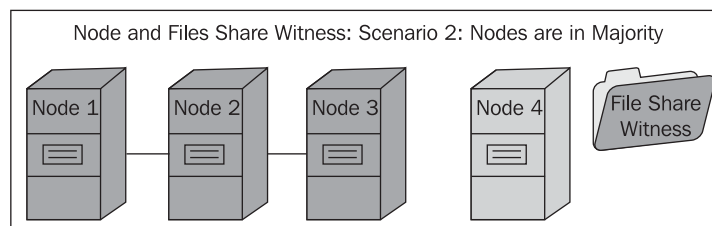
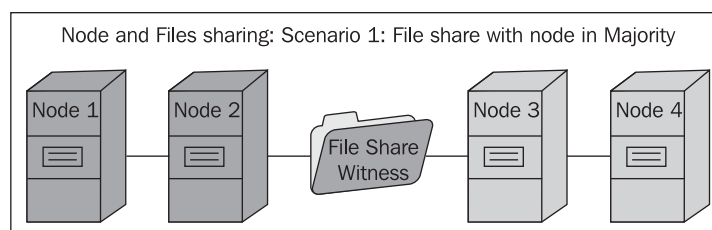


This configuration is recommended where there is a two-node failover cluster or there is an even-node cluster configured. Please note that while configuring an even number of nodes, if the wizard automatically selects Node and Disk Majority, it will also select a witness disk. The wizard also considers selecting the smallest size disk of more than 512MB as a witness disk.

Keep the following things in mind while implementing this configuration:

- Use a small LUN that has a minimum size of 512 MB.
- Make sure that this disk is dedicated to be the disk witness and contains no user data.
- It is not required to assign a drive letter to this LUN; we can decide this based on our cluster need.

- This LUN should be a single volume and should be added as a cluster resource.
 - Ensure that this LUN is validated and configured using hardware RAID and is formatted using the NTFS file system.
- **Node and File share majority:** In this mode, there will be a designated file share that is created as a file share witness. There will be a replica for each of the shares, which is kept on a system disk across all the nodes; however, this is not stored on the file share, which is the core difference between *node and disk majority* and *disk and file share majority*.



This file share keeps an eye on each of the disks so it knows which has the most up-to-date replica. This configuration is recommended in a geographically dispersed failover cluster.

Keep the following in mind while implementing this configuration:

- Use **Server Message Block (SMB)**.
 - Ensure that the file share has a minimum 5 MB of free space.
 - The file share should be dedicated to the cluster.
 - It should not be on a server that is a cluster node or would become cluster node in future – file server would be a good choice.
 - The file share should be on the same forest of the domain.
 - Ensure that the administrator has the full permission for the share and NTFS.
- **No majority – Disk only:** This mode of Quorum is similar to the previous versions of Quorum with Windows Server 2003. This is recommended in only selected configurations. This is because in this case, if a single disk fails, then entire cluster will fail.

The cluster replica will help here as it stores the exact and most up-to-date replica of the cluster database on the shared disk, which is accessible by all the nodes. In case of failure of a single node, this replica is considered as an authoritative replica to help repair or replace the Quorum that has been corrupted or damaged.

Once the configuration for the Quorum is over and we want to have a look at it, there are two options: we may either open a GUI (a cluster management snap-in) or use the command prompt by typing `Cluster / quorum`.

When it comes to making changes to existing Quorum configuration, we may do it using:

- Failover cluster management snap-in
- Managing a cluster
- Action
- More action
- Configuring cluster Quorum settings
- Following the wizard the way we want our Quorum to be configured

Public and Private Network

In a clustering, each cluster node or every node that is a candidate for being a cluster node should have two Network Interface Cards (NICs) as a base requirement – one card can be used to configure the Public Network and second can be used to configure the Private Network.

- **Public Network:** In a clustering environment, a network that is exposed to the public, that is, client systems to connect to is called the Public Network.
- **Private Network:** A network that is used to connect two or more cluster nodes is known as the Private Network, also known as the Heartbeat.

SQL Server uses this Heartbeat network for health detection between two or more nodes, especially in a single-node failover cluster. Consider a situation where we have configured a single-node failover cluster—Node1 and Node2. Node1 is holding ownership of the cluster resources and Node2 is in passive mode and has no access to or ownership of any cluster resources.

In case of failure or unmanaged shutdown, or for any reason if Node1 is rebooted or shut down, the Private Network or Heartbeat between these two nodes fails and the Node2 of the cluster will become the active node. It will now have ownership of and access to all the cluster resources until it fails over to Node1.

SQL Server MVP Brad McGehee, in his article, has mentioned some excellent points that should be considered while configuring the Public and Private Networks. The article can be found at http://www.sql-server-performance.com/articles/clustering/cluster_infrastructure_pl.aspx.

Summary

In this chapter, we have learned about clustering in general and what clustering in Windows and SQL Server means. We have also learned some of the new features introduced with SQL Server 2008.

We learned about Public and Private Networks in clustering, their significance, and how they function in SQL Server clustering. We also looked at some of the things that need to be considered before we go about configuring a network for our cluster nodes. We saw different types of clusters such as Single-Failover Cluster, Multi-Failover Cluster, Multi-site Failover Cluster, and Guest Failover Cluster. We also saw different types of disk storage that can be used while configuring clusters, along with the most important part of a cluster, Quorum.

In the next chapter, we will learn what the prerequisites for SQL Server Cluster installation are and how to install SQL Server Cluster using a wizard and T-SQL.

Where to buy this book

You can buy Microsoft SQL Server 2008 High Availability from the Packt Publishing website: <https://www.packtpub.com/microsoft-sql-server-2008-high-availability/book>

Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our [shipping policy](#).

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.



www.PacktPub.com

For More Information:

www.PacktPub.com/microsoft-sql-server-2008-high-availabilitybook