IDERA® | SQL Server Whitepaper

TOP 5 SQL SERVER **CLUSTER SETUP MISTAKES**

By Kendra Little
Brent Ozar Unlimited™
Microsoft Certified Master, SQL Server

SOLUTION OVERVIEW

Windows Failover Clustering can keep your SQL Server databases online and your customers happy when hardware or operating system problems strike. Failover Clustering also makes your life easier when you need to do maintenance on a physical server: just fail your SQL Server instance over to another server and complete your work with minimal downtime.

But be careful: if you misconfigure your cluster, you're bound for poor SQL Server performance or extended unplanned downtime.

You shouldn't let an otherwise great high availability feature cause weak performance or outages. Let's learn how to avoid the some of the most prevalent SQL cluster setup mistakes.

Need a quick refresher on precisely what a failover cluster is?
Read An Introduction to SQL Server Clusters to get the basics.

TABLE OF CONTENTS

MISTAKE #5: NOT PLANNING YOUR NODES AND INSTANCES

It is essential that you properly plan your nodes and instances. Start by considering the following questions:

How many cluster nodes will you build?

"Nodes" are the servers configured as members of the Windows Failover Cluster. Each node may be part of only one Failover Cluster and must be a member of a Windows domain.

Most people plan Windows Failover Clusters for SQL Servers where both performance and availability is important. If this is the case, plan one node per SQL Server instance plus at least one additional "passive" node.

Ask yourself, "How many servers can fail before performance is degraded?" That answer is the number of passive nodes you need in your cluster.

How many SQL Server instances will you install?

Each SQL Server instance can move from node to node, but each instance can be active on only one node at any given time. Every SQL Server instance has its own collation, user and system databases, logins, and SQL Agent jobs configured, and all those components move along with the instance.

How many Windows Failover Clusters will you manage?

You might be tempted to put all your nodes and SQL Server instances in a single cluster. This seems attractive for cost savings on hardware: a four node cluster with one passive node sounds better than a pair of two node clusters, each with a passive node. But consider the following factors:

1. **SQL Server Licensing Costs:** SQL Server Standard Edition allows you to use two nodes for a single instance in a Windows Failover Cluster. To install the instance on three or more nodes, you need SQL Server Enterprise Edition, which costs significantly more. Software Assurance may be required for the passive node.[i]
2. **Uptime Requirements:** You should apply hotfixes and service packs to Windows and SQL Server on a regular basis. After applying updates, you should test that each SQL Server instance runs successfully on every node. The more nodes and the more instances you have in a cluster, the more downtime each instance will have when you fail it over between nodes.

What kind of storage will you use?

For a Windows Failover Cluster Instance, all nodes must have access to shared storage: each SQL Server user and system database anywhere in the cluster will have just one copy of its data stored in a central location. (Foreshadowing: there's one exception to this. **Read on!**) The shared storage may be a large, sophisticated Storage Area Network (SAN), or a smaller storage array.

What will your staging environment be like?

To support clustered SQL Server in production, you need a non-production SQL Server "Staging" cluster. The staging cluster should match production as closely as possible.

Some Frequently Asked Questions about designing clusters for SQL Server:

- **Default SQL Server Instances:** You may have only one Default SQL Server instance per Windows Failover Cluster. Don't worry too much about that limitation: you may set all instances on a cluster to use port 1433, because each instance will have its own IP address.
- **"Stretch" Clusters in Multiple Datacenters:** Your Windows Failover Cluster can span subnets and datacenters, but you need to have storage replication manage the data transfer[ii]. If you'd like SQL Server to manage the data transfer instead of storage replication, investigate SQL Server Availability Groups. Availability Group instances do not require shared storage and may be combined with Windows Failover Cluster instances. If you do combine the technologies, you will only have automatic failover within the Windows Failover Cluster Instance.[iii]

MISTAKE #4: BUYING THE WRONG HARDWARE

It's easy to get confused when sizing hardware for a clustered SQL Server instance. There's tons of misinformation out there.

"Buy enough memory. But don't assume you're limited to using half of it!"

Don't go cheap on memory: it's critical to SQL Server performance.[iv]

Many people incorrectly believe that if they have two instances on a two node cluster, they need to set SQL Server's Max Server Memory setting to 50% or less of the total memory of each node to protect performance in case both instances need to run on a single node.

Don't do that. On SQL Server 2005, if multiple SQL Server instances share a single node, they will negotiate with each other and balance memory use. You may use the Min Server Memory setting to prioritize one instance over another.[v]

“Buy multiple physical NICs.”

As of Windows Server 2008 and higher, you no longer need to configure a separate “Heartbeat” network for a Windows Failover cluster. Instead, you must make sure that your network path is redundant.

This means you need two physical network adapters (NICs) in each cluster node. You may bring these together into a single network team; the point is that communication must be able to continue if either physical NIC fails.

The NICs must be connected to a network that has redundant switches, and this network is only for your SQL Server and cluster traffic. (In other words, your storage network shouldn’t use this. If you use iSCSI storage, it must have its own dedicated NICs.)[vi]

“Buy the right tempdb storage.”

There’s an exception to the rule mentioned above that all user and system databases must be on shared storage. Beginning in SQL Server 2012, you can configure tempdb on local storage for a clustered SQL Server instance. When performance is important, speeding up tempdb is attractive. Solid State Storage (SSDs) can be expensive to implement in shared storage devices.

Lots of people choose to install SSDs into each server node to make tempdb fast. The SSDs may be either 2.5” SSDs or PCI Express cards. If you take this approach, remember:

- All nodes need identical SSD configurations. When your SQL Server instance fails over and tries to come up on another node, it needs to find tempdb at the right drive letter with the right available capacity.
- Don’t buy just one SSD per node. Failure of a drive under tempdb can be disastrous for a SQL Server instance. Even though you’re using a Windows Failover Cluster you should provision hardware to avoid unplanned failovers as much as possible. If many transactions are “in flight” when a failure occurs, the process of bringing databases online on a new node may take longer than your users like.
- Your staging environment needs the same SSD configuration. Imagine that you’ve started seeing periodic slow performance from your SSDs. A firmware upgrade is available from the vendor. Where do you want to test the firmware upgrade first?

MISTAKE #3: SELECTING THE WRONG VERSION OF WINDOWS OR SQL SERVER

When you're planning a new Windows Failover Cluster, the Windows Server version is important. Selecting the right Windows installation can make your cluster easier to manage, and easier to keep online.

- Windows Server 2008 introduced more robust networking capabilities and critical tools, including the Cluster Validation Wizard.
- Windows Server 2012 brought a major improvement: the cluster determines cluster majority by looking at the current active members of the cluster (not the number of members that were initially configured). This helps keep your SQL Server instance online when failures start to occur.
- Windows Server 2012 R2 added better cluster options to manage witness votes and tie situations.

But don't base your decision only on all the improved features. The most recent Service Pack for Windows Server 2008 R2 was released in March 2011: that's a long time ago! Many fixes have been released that are critical to performance and availability for Windows Failover Clusters. Determining the best patch configuration for Windows Server 2008 R2 becomes increasingly difficult as more time passes.[vii]

Use Windows Server 2012 or 2012 R2 whenever possible

If you're planning a new cluster for SQL Server 2012 and higher, identify the highest Windows Server version supported by your company. If that's not at least Windows Server 2012, map out all the improvements the new OS brings and the hotfixes it helps you avoid: your company may not support a new OS until you advocate for change.

Learn to slipstream if installing SQL Server 2008 or 2008 R2 on Windows Server 2012 and higher

Microsoft supports installing SQL Server 2008 or 2008 R2 on a Windows Server 2012 or 2012 R2 failover cluster. Installation will require extra steps: you must install .NET 3.5 and use a "Slipstream" command line installation to install SQL Server with a Service Pack already in place.[viii]

Don't mix SQL Server Versions

Sometimes people ask if they can install different SQL Server versions (such as a SQL Server 2008 R2 instance and a SQL Server 2012 instance) on a single Windows Failover cluster. This is an uncommon configuration. It's called a "side by side" cluster deployment.

Although this is technically possible, I don't recommend it. Installation is complex: you must install each SQL Server version from lowest to highest when setting up every node.

Even worse, whenever you need to troubleshoot or seek support for an incident, your very specialized configuration makes it more difficult to resolve and find the root cause for the problem.

MISTAKE #2: CONFIGURING QUORUM BADLY

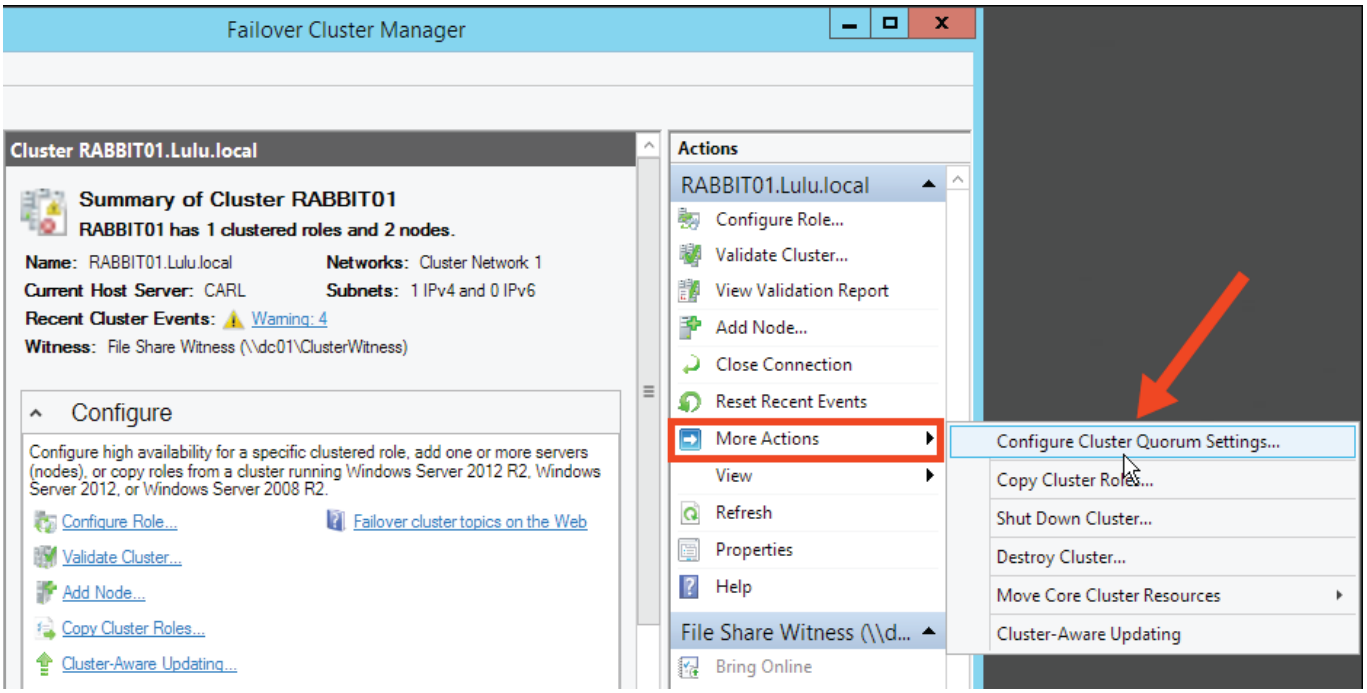
You must understand quorum and configure it properly in the Windows Failover Cluster to keep your SQL Server databases online. This is tricky. The best practices for configuring quorum vary depending on the version of Windows you're using, the number of nodes you have, and the reliability of network communication between the nodes.

Failover clusters want to keep your databases online, but they have a problem: they must keep each SQL Server instance online only on a single node!

"Quorum" is a process by which each member of the cluster stops and takes attendance and checks which members of the cluster own each resource in the cluster. Failover Clustering got a lot better at checking quorum in Windows Server 2012: instead of always counting the initial cluster configuration as the total number of possible votes, at any given time the current members of the cluster are counted as the voting population.

Potential voting members of the cluster are:

- Each node of the cluster
- A witness (optional): This is a disk resource (or a file share) that keeps a copy of critical information about the cluster's state and configuration



The Golden Rule: Every time your cluster configuration changes, re-evaluate quorum to avoid a tie situation

Your cluster wants to always have an odd number of votes available. Clusters just hate a tie, because it's very difficult for them to know what to do!

Imagine you have a Windows Failover cluster with four nodes. You haven't configured a disk witness of any kind. Each node is granted a single vote, so you have a total of four votes. One SQL Server instance is installed on the cluster, and it's currently residing on Node 2.

Suddenly, a network crisis occurs! Nodes 1 and 2 can see one another, but not Nodes 3 and 4. Nodes 3 and 4 can see each other, but neither can see Nodes 1 and 2.

What happens now when each node counts votes? Node 1 and Node 2 can see each other and count two votes out of four: that's not a majority. Node 3 and Node 4 are in the same predicament. Neither group has a majority: that means neither group has the clear right to keep the SQL Server instance online.

Depending on your SQL Server version and whether you've established a Tiebreaker (available in SQL Server 2012R2), the Windows Failover Cluster service may shut down on every node and take your databases offline with it.

Good news: you can decrease the chances of this happening!

Assign a witness to get an odd number of votes

Every cluster may have one witness. It also might not have a witness: it's up to you to decide.

The "witness" is a member of the cluster that helps the cluster decide where your SQL Server Instances should be active. You may choose to set up a witness using one of these configurations:

- A Disk Witness (This is called "Quorum Disk" sometimes)
- A File Share Witness

The Disk Witness may be a little more work to configure, because this disk must be available on shared storage available to all of the nodes. However, the disk witness is a bit more sophisticated than the file share witness and holds more information that may be useful to the cluster when failures occur.

If you decide to use a witness, use a Disk Witness whenever possible.

Here we've got a two node Failover Cluster. To make sure there's an odd number of votes, the administrator configured quorum to use a witness:

Remove votes when appropriate

You can manually remove the vote of a node. In Windows 2012 and Windows 2012 R2, you can do this using the Failover Cluster Manager tool. In Windows 2008 R2 you can do this with PowerShell after applying KB 2494036.[ix]

Removing a vote is useful particularly when you have a cluster that spans datacenters. In this example, Node 1 and Node 2 are in a primary datacenter. Node 3 is in a remote datacenter, and network communication is unreliable. You will only ever manually fail over to the remote datacenter.

In this scenario, you can manually remove the vote from the node in the remote datacenter and use a witness to achieve an odd number of votes in the primary datacenter.

Windows Server 2012 "Dynamic Quorum" dynamically adjusts node votes

The "Dynamic Quorum" feature in Windows 2012 makes life easier. When this is enabled, the cluster will count up current members of the cluster and decide if it should dynamically remove a vote so that you have an odd number.

You can check dynamic weights in Windows Server 2012 using PowerShell. In this four node cluster, I have allowed all four nodes a vote ("NodeWeight"), but the cluster has dynamically reduced the vote for the node ELVIS. This results in an odd number of total votes.

```
PS C:\> Get-ClusterNode | Format-Table name, dynamicweight, nodeweight, id, state -AutoSize

Name    DynamicWeight NodeWeight Id State
----    ------------- ---------- -- -----
CARL                1          1 2  Up
CHARLIE             1          1 3  Up
ELVIS               0          1 1  Up
ROGER               1          1 4  Up
```

Node Dynamic Weight is visible in the graphical interface of the Windows Failover Cluster Manager beginning in Windows Server 2012 R2.

Windows Server 2012 R2: Dynamic counting of witness votes; tiebreaker option

You've got even more options in Windows Server 2012R2: the cluster can decide whether or not it should give the witness a vote based on the state of witness and the available number of nodes. It also provides more options, helping nodes in your primary datacenter stay online if you end up in a tied-vote situation between your primary and disaster recovery datacenters.

Rules for designing quorum

Here are a few rules to keep you sane when setting up your cluster:

- Your goal is to have an odd number of votes: use a witness when necessary
- In Windows Server 2012 R2, always use a witness and allow the cluster to dynamically manage the vote
- If your cluster stretches across sites and network communication is unreliable, consider removing votes from nodes in the secondary site
- Use a Disk Witness whenever possible (instead of a File Share Witness)
- Before going live, test out many failure scenarios to understand how your version of Windows handles dynamic voting.

MISTAKE #1: SKIPPING CLUSTER VALIDATION (OR IGNORING WARNINGS)

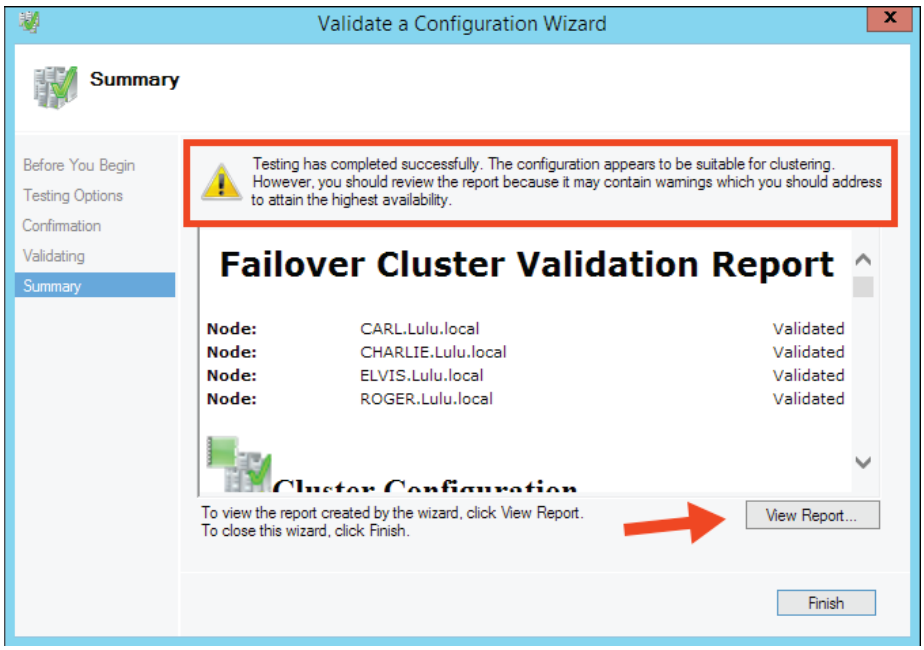
Cluster Validation is the most important part of building and configuring a Windows Failover Cluster. Never skip this step. You should run validation a few times:

- Validate the configuration prior to cluster install
- Run validation after the cluster is set up and you've completely installed and configured your SQL Server instances on the cluster
- Validate whenever you make changes to the cluster

Not all validation tests can be run online. Make sure validation results are perfect before going live to avoid having to schedule a downtime.

Take every warning or error seriously

Validation is the main way your cluster can warn you that about problems in the configuration. Never assume that warnings can be skipped.



Before you take the cluster into production use, make sure that you've run a full cluster validation and corrected every error and warning. If you do not correct a warning, document exactly why it is ignorable. Save a copy of your cluster validation reports off in a safe spot in case you need them for reference when trouble strikes.

What if you didn't build the cluster?

Sometimes the Database Administrator who manages the cluster isn't the same person who chose the hardware, installed the Windows Failover Cluster Feature, and added the nodes to the cluster. Perhaps the cluster was handed off to you and you were told, "The cluster's all ready."

Translation: What this really means is that it's time for you to make sure that Cluster Validation is perfect.

YOU CAN AVOID THESE SETUP MISTAKES!

Let's face it: hardware will fail. Windows Failover Clustering is a great technology that can keep your SQL Server instances and databases online and working when the inevitable happens.

Armed with the knowledge in this paper, you can set your next SQL Server Failover Cluster the right way.

Footnotes

i. See "Licensing for High Availability" in the Microsoft SQL Server 2012 and 2014 Licensing Guides:

2012: http://download.microsoft.com/download/7/3/C/73CAD4E0-D0B5-4BE5-AB49-D5B886A5AE00/SQL_Server_2012_Licensing_Reference_Guide.pdf

2014: http://download.microsoft.com/download/6/6/F/66FF3259-1466-4BBA-A505-2E3DA5B2B1FA/SQL_Server_2014_Licensing_Datasheet.pdf

ii. See "SQL Server Multi-Subnet Clustering":
http://msdn.microsoft.com/en-us/library/ff878716.aspx

iii. See "Failover Clustering and AlwaysOn Availability Groups":
http://msdn.microsoft.com/en-us/library/ff929171.aspx

iv. For example builds, see "What's the Smallest SQL Server You Should Build":
http://www.brentozar.com/archive/2014/01/whats-the-smallest-sql-server-you-should-build

v. See "Running Multiple Instances of SQL Server"
http://msdn.microsoft.com/en-us/library/ms178067.aspx.

Note that for many configurations, allowing only 1-2GB of physical memory to be free may not be sufficient for the operating system.

vi. Hardware Requirements for Failover Clusters:
http://technet.microsoft.com/en-us/library/cc771404.aspx

vii. Notable KBs for Windows 2008 R2 include: KB 2566205, KB 2468345, KB 2545635, KB 2553549

viii. For more information on Slipstreaming:
http://blogs.msdn.com/b/petersad/archive/2011/07/13/how-to-slipstream-sql-server-2008-r2-and-a-sql-server-2008-r2-servicepack-1-sp1.aspx

ix. KB 2494036:
http://support.microsoft.com/kb/2494036

TRY IDERA'S SQL Diagnostic Manager

24X7 SQL PERFORMANCE MONITORING, ALERTING AND DIAGNOSTICS

- **Performance monitoring** for physical and virtual SQL Servers
- **Deep query analysis** to identify excessive waits and resource consumption
- **History browsing** to find and troubleshoot past issues
- **Adaptive & automated alerting** with 100+ pre-defined and configurable alerts
- **Capacity planning** to see database growth trends and minimize server sprawl
- **SCOM management pack** for integration with System Center

Try it FREE for 14 days.

START FOR FREE