The Digital Ranch

Cloud Project: Documentation

By: Steve Durham, Tyson Harper, Kyle Holbrook, & Trevor Schow
# Table of Contents

Introduction ..................................................................................................................................... 4

Preliminary Diagrams of Project Architecture ............................................................................... 5
  Domain ........................................................................................................................................ 5
  IIS 7 Cluster ................................................................................................................................ 5
  SQL 2008 Failover Cluster ......................................................................................................... 6

Creation and Installation ................................................................................................................. 7
  Virtual\ Physical Machines ......................................................................................................... 7
    MS Server 2008 Installation ..................................................................................................... 8
    Domain Controller .................................................................................................................. 10
  IIS 7 Web Server Farm ............................................................................................................. 11
    IIS 7 Server Install ................................................................................................................ 11
    IIS Web Farm Framework Installation ................................................................................. 14
    Website Installation .............................................................................................................. 18
  Network Load Balancer Install ............................................................................................. 18
  SQL Cluster .............................................................................................................................. 20
    Carve out the shared disk ....................................................................................................... 20
    Present the shared disks to the cluster nodes ....................................................................... 20
    Adding Windows Server 2008 Application Server Role ..................................................... 24
    Adding the Failover Cluster Feature ..................................................................................... 28
    Running the Windows Server 2008 Validate Cluster Configuration ................................... 31
    Creating the Windows Server 2008 Cluster ........................................................................ 35
    Configuring your cluster quorum .......................................................................................... 37
    Installing and Configuring MSDTC .................................................................................... 40
    Installing SQL Server 2008 Standard on a Windows Server 2008 cluster ....................... 44
    Adding a node on a SQL Server 2008 Failover Cluster ....................................................... 61
    Witness SQL Server Install (SQL-4) .................................................................................... 66

Testing and Proving Functionality ................................................................................................ 80
  Phase 1: Ensure that IIS 7 cluster nodes are syncing properly ................................................. 81
    Load the default IIS 7 website in IIS-Primary ...................................................................... 81
    Alter Webpage on IIS-Secondary ......................................................................................... 84
    Load DentistPackage website onto IIS-Primary ................................................................... 84
Phase 2: Ensure that SQL Server 2008 cluster nodes are syncing properly ......................... 85
  Loading DentistPackage Database on SQL-1 & SQL-2 ....................................................... 85
  Making Changes to the SQL Databases on each node ...................................................... 86
Phase 3: Testing redundancy, outages, and uptime .............................................................. 88
  Test 1: Removing IIS 7 Cluster Nodes ............................................................................ 88
  Test 2: Removing SQL Cluster Nodes ............................................................................ 96
  Test 3: Adding Non-configured Servers to the IIS Cluster Farm ................................... 102
Conclusion of Tests ........................................................................................................... 108
Final Report/Recommendations ......................................................................................... 109
  IIS 7 Cluster .................................................................................................................... 109
  SQL 2008 Failover Cluster ............................................................................................. 110
Appendix ............................................................................................................................. 111
  Email 1: .......................................................................................................................... 113
  Email 2: .......................................................................................................................... 113
  Email 3: .......................................................................................................................... 113
Introduction

The Cloud Project is to create a Microsoft Windows IIS 7 Cloud (farm) and a SQL 2008 R2 Failover Cluster. This project consists of four main parts:

1. Diagramming and documenting the architecture of the plan
2. Creation and installation of the IIS and SQL clouds using The Digital Ranch’s equipment (hardware and software)
3. Test the Cloud (prove full functionality)
4. Create reports and documentation regarding the overall findings and success of the project.

The Digital Ranch will provide all hardware and software necessary – including virtual machines. Part of this project, namely the IIS 7 servers, will be virtual. It is understood that for this proof of concept that the virtualization might not be optimal, but it will be sufficient for demonstrating the capabilities of the cloud and providing full documentation.

The IIS 7 webservers will run websites and the content for the websites will be stored and handled by the SQL servers. The goal of the project is to have zero downtime, meaning webservers and database servers can fail completely while still maintaining 100% uptime for the entire cloud as a whole.
Preliminary Diagrams of Project Architecture

These diagrams and descriptions are preliminary and are provided to give a visual representation of the Cloud Project. It will be desired to diagram and document the real set-up and implementation in a similar way. Use these diagrams to set-up the Cloud’s computer names, IP Addresses, and other details that are provided in this document. Provide all other important details in the final diagrams and documentation.

Domain
The IIS 7 and SQL Clouds will all run on domain machines. A new domain (cloud.local) will need to be created. It is also important to create the domain controller VM and the domain. All machines, physical and virtual, must be members of the domain.

IIS 7 Cluster
The IIS 7 Cluster will consist of three servers, all virtualized. The computer names are IIS-1, IIS-2, and IIS-3. IP Addresses and OS specifications are shown below. The Digital Ranch will provide all content for the websites and associated databases.
SQL 2008 Failover Cluster

The SQL 2008 Failover Cluster will consist of three servers, two physical and one virtual. The computer names for the physical servers are SQL-1 and SQL-2. The virtual server is SQL-3. IP Addresses and OS specifications are shown below. The virtual server will function as the “witness” or “watchdog” server. It is important to create the virtual machine and also install and configure the OS and all software on the physical servers.

The SQL Servers will run in Failover Clustering. The databases will be stored on an iSCSI disk. The Digital Ranch will set-up the iSCSI disk and will provide all content for the databases.
Creation and Installation

Virtual\ Physical Machines

To provide a proper cloud structure a solution involving any number of separate physical computers is recommended. If virtual machines are in a cluster there is one link that can stop them all, the host computer. However, for this proof of concept, we used a combination of physical and virtual computers in order to most efficiently deploy and test the model given by The Digital Ranch.

ZION server information

A rack server named ZION, outfitted with an Intel Xeon processor and 8 GB of memory housed our virtual machines. We used Windows Server Hyper-V to deploy and manage our virtual machines. Those machines were:

- Active Directory Domain Controller
- Three IIS 7 web servers
- One SQL Server virtual machine which was later left behind due to project changes
For the SQL Server 2008 R2 cluster, we use two separate machines, both of relatively modest specifications. They have Intel Pentium 4 processors and 5 GB of memory, which both run the program as well as the clustering utilities without any problems.

**MS Server 2008 Installation**

ZION runs Windows Server 2008 R2 Datacenter Edition. The virtual machines are outfitted with Windows Server 2008 R2 Enterprise Edition, a requirement for clustering. Each machine is allocated 1024 MB of memory and 20GB of hard drive space to have resources needed to carry out clustering and server operations. As can been seen in the following figure, ZION houses the four IIS 7 virtual machines: IIS-1, IIS-Primary, IIS-Secondary. For convenience we also have the Active Directory Domain Controller, DC-1, housed in ZION.
For the two SQL Server 2008 machines, SQL-1 and SQL-2, we chose Windows 2008 R2 Enterprise Edition, again because that version is required for clustering. Since we were only clustering two instances of SQL Server 2008 R2, upgrade to the Enterprise Edition of that software was not required. Using SQL Server 2008 R2 Enterprise Edition allows for many more machines to join the cluster.

**SQL-1 System Information**

![Windows Server 2008 R2 Enterprise Edition](image)
Domain Controller

An Active Directory Domain Controller that we named DC-1 is used with our computers in order to more easily administer them. Since there are only six computers in our forest, as well as a light workload for the domain controller, a virtual machine suffices for our needs.

DC-1 Role Information

DC-1 System Information
IIS 7 Web Server Farm
We will be installing IIS7 on all the servers that will be part of webserver cluster. Once installed we will proceed to install the Microsoft Web Farm Framework and finally add the IIS7 instances to the farm.

IIS 7 Server Install
Requirements
- Confirm you have administrative rights
- For ease of setup, add cluster computers to the domain
- Network settings, such as IP, have been configured
- Updates have been installed

Installation
IIS is a server role that can be installed through the Server Manager interface.

1. Click Start -> All Programs -> Administrative Tools -> Server Manager
2. Select Roles. From the Server Manager Menu Select Action -> Add Roles
   This will open up the Roles Wizard. Click Next

   ![Select Server Roles](image)

   (select Web Server IIS)

3. Check Web Server (IIS). Click Next, Next
4. Select Additional Roles for the Web Server. Click Next
(select additional components such as ASP.NET, etc)

5. Confirm Installation settings. Click Install
6. Confirm your installation by visiting http://localhost/
7. Repeat for each IIS Server to be included into the cluster

**IIS Web Farm Framework Installation**

**Requirements**
- Installation of IIS, ASP.NET, other core components

**Structure Design**
IIS-1 will act as the farm controller with IIS-Primary and IIS-Secondary acting as nodes in the farm.

**Microsoft IIS7 Web Farm**

(web farm design)
**Installation**
Install Microsoft Web Farm Framework onto IIS-1.

1. Download and install Web Farm Framework at [http://go.microsoft.com/?linkid=9739157](http://go.microsoft.com/?linkid=9739157). This is installed on your Farm Controller server or in the case of the example, IIS-1. It can be installed on any machine running version IIS7 or higher.

2. Once the Web Farm Framework is installed, a new node will appear in the left navigation tree called Server Farm. Right click Server Farm and you will have the option to Create Server Farm…

3. After selecting create, you will be presented with the following dialog window. Here you will give the farm a name, in this case we called it cloudfarm. Check “Provision Server Farm”. This will automatically provision all of the servers in the farm. You can also select the load balancing option to make it available for load-balancing with the IIS Application Request Routing (ARR) load-balancer.
4. The next window allows us to add servers to the farm. First add IIS-Primary and check “Primary Server”. Then add IIS-Secondary. Once you click finish the farm will automatically provision the management software to the nodes in the farm.
(add IIS-Primary to Farm)

(Add IIS-Secondary to Farm)
Website Installation

When installing the website, all content and code is loaded to the IIS-Primary. Anything that is loaded to IIS-Primary will replicate to any secondary servers. In the case of this setup the website code would automatically copy over to IIS-Secondary. The Farm will also replicate any platform components installed.

Network Load Balancer Install

The purpose of Web Farm Framework, installed on IIS-1, is to keep the servers in sync and provision new servers added to the farm. We will now install the Network Load Balancer which will monitor and distribute traffic to the servers in the farm. We will first install the NLB, set IP and designate the servers to be included in the Balance.

There are few different methods to installing the NLB.

- Using Server Manager, just click Add Feature and then select Network Load Balancing
- Using windows server 2008 / R2 command line, by typing “ocsetup NetworkLoadBalancingFullServer”
- Use ServerManagerCmd! From a command line, type “servermanagercmd –install nlb”

We will be using the first option by using the Server Manager interface.

1. Click start -> Administrative Tools ->Server Manager
2. Select Features. From the Server Manager Menu Select Action -> Add Feature
   This will open up the Features Wizard.
3. Select the Network Load Balancing checkbox and click next
(Select Network Load Balancing)

4. Once you complete installation of NLB, check the properties of your network adapter for NLB option.
5. Open Network Load Balancing Manager, Right-click **Network Load Balancing Clusters**, and then click **New Cluster**.

6. Our Next step is to Add the Nodes/servers. Add your first Host either by name or IP. In this example we are adding IIS-Primary. Next select the Interface Name.

7. Select Priority. Priority specifies a unique ID for each host. The host with the lowest numerical priority among the current members of the cluster handles all of the cluster’s network traffic that is not covered by a port rule. Click Next, Next,

8. In **Cluster Parameters**, select values in **IP Address and Subnet mask** (for IPv6 addresses, a subnet mask value is not needed). Type the full Internet name that users will use to access this NLB cluster. In our example we are using 10.10.10.204 as our NLB address.

9. Leave Cluster operations mode to Unicast. Click Next.

10. To Add more Hosts Right Click the Cluster and select add new host to Cluster

---

**SQL Cluster**

*Carve out the shared disk*

In preparation for running SQL Server 2008 on this cluster, it is recommended to create at least 3 disks - one for the quorum disk, one for MSDTC, and one for the SQL Server system databases. The quorum and MSDTC disks can be as small as 1GB, although Microsoft TechNet specifies a 512MB minimum for the quorum disk. For this project there are 3 shared disks:

- iqn.2011-02.PORKCHOP:iporkchop
- iqn.2011-02.PORKCHOP:iporkchop2
- iqn.2011-02.PORKCHOP:iporkchop3

If iSCSi is used as the shared storage in a production environment, a dedicated network should be used so as to isolate it from all other network traffic. This also means having a dedicated network card on your cluster nodes to access the iSCSI storage.

*Present the shared disks to the cluster nodes*

Windows Server 2008 comes with iSCSI Initiator software that enables connection of a Windows host to an external iSCSI storage array using network adapters. Thistoolcan be launched from Administrative Tools and select iSCSI Initiator.
To connect to the iSCSI target:

1. In the **iSCSI Initiator Properties** page, click on the **Discovery** tab.

2. Under the **Target Portals** section, click on the **Discover Portal** button.
3. In the **Discover Target Portal** dialog, enter the DNS name or IP address of your iSCSI Target and click **OK**. For this project the IP address is **10.10.10.26**. If the target is hosted on another Windows host as an image file, make sure that the Windows Firewall is configured to enable inbound traffic to port 3260. Otherwise, this should be okay.

4. Back in the **iSCSI Initiator Properties** page, click on the **Targets** tab. There should be a list of the iSCSI Targets that we have defined earlier and that were previously partitioned. If not try clicking the **Refresh** button if needed.
5. Select one of the targets and click on the Connect button.
6. In the Connect to Target dialog, select the Add this connection to the list of Favorite Targets checkbox. Click OK.
7. Once that is done, the status of the target should be changed to **Connected**. Repeat this process for all the target disks that were initially created on both of the servers that will become nodes of the cluster (SQL-1 and SQL-2 for this project).

Once the targets have been defined using the iSCSI Initiator tool, the disks can now be brought online, initialized, and create new volumes using the Server Manager console. After the disks have been initialized and volumes created, try logging in to the other server and verify that all the disks are there as well. Rescan the disks if they haven’t yet appeared.

**Adding Windows Server 2008 Application Server Role**

Since SQL Server 2008 will be installed later on, add the Application Server role on both of the nodes (SQL-1 and SQL-2). A server role is a program that allows Windows Server 2008 to perform a specific function for multiple clients within a network. To add the Application Server role,

1. Open the **Server Manager** console and select **Roles**.
2. Click the **Add Roles** link. This will run the **Add Roles Wizard**.

3. In the **Select Server Roles** dialog box, select the **Application Server** checkbox. This will prompt for the additional features required for Application Server role. Click **Next**.
4. In the Application Server dialog box, click Next.
5. In the Select Role Services dialog box, select **Incoming Remote Transactions** and **Outgoing Remote Transactions** checkboxes. These options will be used by MSDTC. Click Next.

6. In the **Confirm Installation Selections** dialog box, click **Install**. This will go thru the process of installing the Application Server role.
7. In the **Installation Results** dialog box, click **Close**. This completes the installation of the Application Server role on the first node (SQL-1). This process will have to be repeated for the other server (SQL-2).
Adding the Failover Cluster Feature

Windows Server 2008 calls them features which are simply software programs that can support or augment the functionality of server roles. Failover clustering simply augments the role as an application server by making it highly available. It is disabled by default, and must be added on both of the servers that will form a part of the cluster (SQL-1 and SQL-2).

To add the Failover Clustering feature:

1. Open the **Server Manager** console and select **Features**.
2. Click the **Add Features** link. This will run the **Add Features** Wizard
3. In the **Select Features** dialog box, select the **Failover Clustering** checkbox and click **Next**.

4. In the **Confirm Installation Selections** dialog box, click **Install** to confirm the selection and proceed to do the installation of the Failover Clustering feature.
5. In the **Installation Results** dialog box, click **Close**. This completes the installation of the Failover Clustering feature on the first node (SQL-1).
That's how simple and easy it is to add the Failover Clustering feature in Windows Server 2008. This will have to be done on both nodes to complete the process. Once the Failover Cluster Feature is installed on both nodes (SQL-1 and SQL-2), the next step is to validate the servers to see if they are ready for clustering.

**Running the Windows Server 2008 Validate Cluster Configuration**

The Failover Cluster Configuration Program allows for the acquisition of hardware for Windows Server 2008 Failover Clustering very easy and simple. Basically, the hardware will be supported for clustering if it meets these two requirements: the server has a “Certified for Windows Server 2008” logo and it passes this wizard.

**One word of caution**: do not skip any error message that this wizard generates in the final report. Doing so would simply mean that the configuration going forward will be unsupported. However, the cluster will still be valid and work with warning messages but caution should be exercised. This wizard only needs to be run on either of the nodes, but there is no harm in running it on both nodes (SQL-1 and SQL-2).

To run the **Validate Cluster Configuration** Wizard:

1. Open the **Failover Cluster Management** console
2. Under the **Management** section, click the **Validate a Configuration** link. This will run the **Validate a Configuration** Wizard.

3. In the **Select Servers or a Cluster** dialog box, enter the hostnames of the nodes that you want to add as members of your cluster and click **Next**. For this project the host names are SQL-1 and SQL-2.
4. In the **Testing Options** dialog box, click **Next** to run all the necessary tests to validate whether or not the nodes are OK for clustering. If this is the first time to run the wizard, all the tests for validation must be run. For succeeding runs, only selectively chosen tests need to be run to validate previous warnings/problems with the cluster.
5. In the **Confirmation** dialog box, click **Next**. This will run all the necessary validation tests.

![Confirmation dialog box](image1)

6. In the **Summary** dialog box, verify that all the report returns successful.

![Summary dialog box](image2)

The wizard will say whether or not it recommends that it can proceed to the next step of creating the cluster. As mentioned earlier, do not attempt to go any further if this report returned any error messages. The bottom line is simply to make sure that the report returns a successful validation before creating the cluster.

![View Report](image3)
Creating the Windows Server 2008 Cluster

It is now time to create the Windows Server 2008 cluster. It's as easy as running the Create Cluster Wizard on either of the nodes. Make sure that the virtual hostname and IP address are ready before proceeding.

To run the Create a Cluster Wizard:
1. Open the Failover Cluster Management console
2. Under the Management section, click the Create a Cluster link. This will run the Create Cluster Wizard
3. In the Select Servers dialog box, enter the hostnames of the nodes to be added as members of the cluster and click Next.
4. In the **Access Point for Administering the Cluster** dialog box, enter the virtual hostname and IP address that will be used to administer the cluster. Click **Next**.

5. In the **Confirmation** dialog box, click **Next**. This will configure **Failover Clustering** on both nodes of the cluster, add DNS and Active Directory entries for the cluster hostname.

6. In the **Summary** dialog box, verify that all the report returns successful.

To validate whether the cluster is working or not a simple test would be to do a continuous PING on the virtual hostname or IP address that has been assigned to the cluster.
Configuring your cluster quorum

This section is sometimes necessary especially when Windows Server 2008 decides to take a different disk subsystem as a quorum other than the one originally intended. Notice that in the Create a Cluster wizard, there was no option to select the disk subsystem that can be used as a quorum disk (now called the "witness" disk). By default, the Create a Cluster wizard will use the first available disk as the witness disk. To validate, check the Storage node under the Failover Cluster Management console.

To configure the quorum in a failover cluster:

1. Open the Failover Cluster Management console
2. Select the name of the cluster that was just created. Right-click on the cluster, select More Actions, and click Configure Cluster Quorum Settings... This will open up the Configure Cluster Quorum Wizard
3. In the **Select Quorum Configuration** dialog box, select the **Node and Disk Majority (recommended for your current number of nodes)** option. The options presented will depend on how the cluster should be configured. The current selection is for a 2-node cluster configuration.
4. In the **Configure Storage Witness** dialog box, validate and select the disk that the cluster will use as the quorum/witness disk. Click **Next**

5. In the **Confirmation** dialog box, verify that the disk configuration for the quorum/witness disk is correct. Click **Next**
6. In the **Summary** dialog box, verify that all the configurations are successful.

**Installing and Configuring MSDTC**

The Microsoft Distributed Transaction Coordinator (MSDTC) is a transaction manager that permits client applications to include several different data sources in one transaction and which then coordinates committing the distributed transaction across all the servers that are enlisted in the transaction. SQL Server uses the MSDTC service for distributed queries and two-phase commit transactions, as well as for some replication functionality.

Windows Server 2008 made it simpler to configure MSDTC by providing a more straightforward process with fewer steps and less configuration.

To install and configure MSDTC:

1. Open the **Failover Cluster Management** console on any of the cluster node.
2. Under the cluster name, right-click on **Server and Applications** and select **Configure a Service or Application**. This will run the **High Availability Wizard**

3. In the **Service or Application** dialog box, select **Distributed Transaction Coordinator (DTC)** and click **Next**.
4. In the **Client Access Point** dialog box, enter the name and IP address of the clustered MSDTC. This should be a different IP addresses and host name from the one that the Windows Server 2008 cluster is already using. Click **Next**.
5. In the **Select Storage** dialog box, select the disk subsystem that will be used by MSDTC. These disk subsystems have to be defined as available storage in your cluster. In the example below, **Cluster Disk 2** was used and **Cluster Disk 3** was left for SQL Server later in the installation process. Click **Next**

6. In the **Confirmation** dialog box, validate the configuration you have selected for MSDTC and click **Next**
7. In the **Summary** dialog box, click **Close**. This completes the installation of MSDTC on the cluster.

To validate the installation of MSDTC expand the **Services and Applications** node and check the cluster name of MSDTC. Make sure that all of the dependency resources are online.
Installing SQL Server 2008 Standard on a Windows Server 2008 cluster

.NET Framework 3.5 with Service Pack 1 and Windows Installer 4.5 should both already be installed and are a prerequisite to continue installing SQL Server 2008 Standard.

Note: SQL Server 2008 Standard Edition can only support a two node cluster. For clusters that are greater than two nodes, a different version of SQL Server 2008 is needed.

To install SQL Server 2008:

1. Run `setup.exe` from the installation media to launch SQL Server Installation Center. Click on the Installation link on the left-hand side.
2. Click the New SQL Server failover cluster installation link. This will run the SQL Server 2008 Setup wizard.
3. In the Setup Support Rules dialog box, validate that the checks return successful results and click Next.
4. In the **Product Key** dialog box, enter the product key that came with the installation media and click **Next**.
5. In the **License Terms** dialog box, click the **I accept the license terms** check box and click **Next**.
6. In the **Setup Support Rules** dialog box, click **Install**. Validate that the checks return successful results. If the checks returned a few warnings, make sure to fix them before proceeding with the installation.

For the Windows Firewall, make sure that the appropriate port number is open on which SQL Server will communicate. This can also be done after the installation. Alternatively, disable Windows Firewall during the installation and enable it later with the proper configuration. Click **Next** to proceed.
7. In the **Feature Selection** dialog box, select only the components that need to be installed. For the **Shared feature directory**, keep the default path if there is sufficient disk space on the C:\ drive or anywhere that is a **local disk** as this will be used by the SQL Server installation process later on. The directory for the clustered database engine will be different. Click **Next**.
8. In the **Instance Configuration** dialog box, enter the SQL Server Network Name. This is the name that will be available on the network for the clients. This will vary depending on the selection of whether it is a default or named instance. For this project, Named instance is selected.

A couple of things need highlighting in this section. By default, the instance name is used as the **Instance ID**. This is used to identify installation directories and registry keys for the instance of SQL Server and is helpful when to running multiple instances in a cluster. This is the case for default instances and named instances. For a default instance, the instance name and instance ID would be **MSSQLSERVER**. To use a non-default instance ID, select the **Instance ID** box and specify a value.

The section on **Detected SQL Server instances and features on this computer** would make sense if there are other SQL Server instances running on the server.
9. In the **Disk Space Requirements** dialog box, check that there is enough space on the **local disks** to install the SQL Server 2008 binaries and click **Next**.
10. In the **Cluster Resource Group** dialog box, check the resources available on the Windows Server 2008 cluster. This will tell if a new Resource Group will be created on the cluster for SQL Server. To specify the SQL Server cluster resource group name, either use the drop-down box to specify an existing group to use or type the name of a new group to create it. Click **Next**.
11. In the **Cluster Disk Selection** dialog box, select the available disk groups that are on the cluster for SQL Server 2008 to use. Click **Next**.
12. In the **Cluster Network Configuration** dialog box, enter the IP address and subnet mask that the SQL Server 2008 cluster will use. Deselect the checkbox under the **DHCP** column as it will be using static IP addresses.
13. In the **Cluster Security Policy** dialog box, accept the default value of **Use service SIDs (recommended)**.
14. In the **Server Configuration** dialog box, enter the credentials that will be used for the SQL Server service accounts in the **Service Accounts** tab. In the **Collation** tab, select the appropriate collation to be used by SQL Server. Note that the startup type is set to manual for all cluster-aware services and cannot be changed during the installation process. Click **Next**.
15. In the **Database Engine Configuration** dialog box, select the appropriate **Authentication Mode**. To add the currently logged on user to be a part of the SQL Server administrators group, click the **Add Current User** button.
On the Data Directories tab, enter the path where the system and user database files will be created. This will default to the first shared disk in the cluster so in case changes need to occur, change it to the other shared disks to be used by SQL Server 2008, modify accordingly. Click Next.

16. In the Error and Usage Reporting dialog box, click Next.
17. In the **Cluster Installation Rules** dialog box, verify that all checks are successful and click **Next**.
18. In the **Ready to Install** dialog box, verify that all configurations are correct. Click **Next**.
19. In the **Complete** dialog box, click **Close**. This concludes the installation of a SQL Server 2008 Failover Cluster.

At the completion of a successful installation and configuration of the node, a fully functional failover cluster instance has been created. To validate, open the **Failover Cluster Management** console, and click on **SQL Server (TDRSQL1)** under **Services and Applications**. Make sure that all dependencies are online.
Although a fully functioning SQL Server 2008 failover cluster has been created, it does not have high-availability at this point in time because there is only one node in the failover cluster. It is necessary to add the second node to the SQL Server 2008 cluster.

**Adding a node on a SQL Server 2008 Failover Cluster**

Now that a working failover cluster has been created, make it highly available by adding nodes. The number of nodes that can be added in a failover cluster depends on the editions of SQL Server that is used. A Standard Edition of SQL Server 2008 can support up to two (2) nodes in a failover cluster while the Enterprise Edition supports up to sixteen (16) nodes, which is practically the limit for the Enterprise Edition for Windows Server 2008.

To add a node on a SQL Server 2008 failover cluster:
1. Run `setup.exe` from the installation media to launch SQL Server Installation Center
2. Click on the **Installation** link on the left-hand side. Click the **Add node to a SQL Server failover cluster** link. This will run the SQL Server 2008 Setup wizard.

Note: If any pop ups appear click through them as they are just warning that should go away once the patches are installed after the installation of SQL.
3. In the **Setup Support Rules** dialog box, validate that the checks return successful results and click **OK**.

4. In the **Setup Support Rules** dialog box, click **Install**. Validate that the checks return successful results. Again, make sure to fix any errors returned by this check before proceeding with the installation.
5. In the **Product Key** dialog box, enter the product key that came with the installation media and click **Next**.

6. In the **License Terms** dialog box, click the **I accept the license terms** check box and click **Next**.
7. In the **Cluster Node Configuration** dialog box, validate that the information for the existing SQL Server 2008 cluster is correct.

8. In the **Service Accounts** dialog box, verify that the information is the same as what was used to configure the first node.
9. In the Error and Usage Reporting dialog box, click Next
10. In the Add Node Rules dialog box, verify that all checks are successful and click Next
11. In the Ready to Add Node dialog box, verify that all configurations are correct and click Install
12. In the Complete dialog box, click Close. This concludes adding a node to a SQL Server 2008 Failover Cluster

Validate the cluster installation by expanding the Services and Applications node and check the cluster name of the SQL Server instance. There should be an option to move the service to another node, in this case, the node that was just added in the failover cluster.
Witness SQL Server Install (SQL-4)

1. Click on the Installation hyperlink on the left hand side of the screen. Then click on the New Server stand-alone installation link on the right side of the screen.
2. Verify that no checks have failed. If any checks have failed, click on the **Show details** button or **View detailed report link** to find out the cause, correct it, then click on the **Re-run** button to perform the checks again.
3. Click **Install** if the Setup Support Files need to be installed.

4. Click **Next** to finish installing the Setup Support Files.
5. Once all checks have passed and the Setup Support Files have been installed, the option to select the edition and to enter the product key will appear. Click **Next** after the product key has been entered.

6. Click in the **I accept the license terms** check box, then click on the **Next** button.
7. Select the features that should be installed then click on the Next button. (For this example the defaults were used.)

8. Click Next
9. Click **Next** again after the Installation Rules have been tested.

10. Verify that the Named Instance is set to the appropriate name and click the **Next** button.
11. This screen checks to make sure there is sufficient disk space on the drive that will be used for the install. Click Next.
12. The next two screen shots allow for the set up of the service accounts that will be used to run SQL Server. Be sure to add the administrator and set the passwords during this step in the Server Configuration. Click **Next**.
13. The Authentication Mode should be set to **Windows authentication mode** and be sure to click the **Add Current User** button in order to add the current user. Click **Next**.

14. Again, click the **Add Current User** button in order to add the current user. Click **Next**.
15. Reporting Services Configuration, select the **Install the native mode default configuration** and then click **Next**.

16. This screen simply asks how to report and send error information to Microsoft and can safely be skipped if no information is need to be shared. Click the box if Microsoft help is wanted. Click on **Next** again.
17. Click **Next** again after the Installation Configuration Rules.

18. Ready to Install, this screen summarises what is about to installed and gives a last chance for any changes that need to happen. Click on **Install** to start the installation process.
19. SQL Server will now install. Click Next when prompted.
Installation Progress

- Setup Support Rules
- Product Key
- Licence Terms
- Feature Selection
- Instance Configuration
- Disk Space Requirements
- Server Configuration
- Database Engine Configuration
- Error and Usage Reporting
- Installation Rules
- Ready to Install
- Installation Progress
- Complete

Scheduling package installation.
20. After the install the following screen will appear. It may be worth click on the installation log at the top of the screen to check that everything has gone as expected.
Testing and Proving Functionality

You will need to come up with your own tests to prove and ensure full functionality. The tests must prove zero downtime without any delays. Also, the tests need to show that all sites in the IIS 7 cluster are mirrored properly across all three servers and that sessions do not get lost when a node (server) fails.

A couple of example tests would be:

- Pull nodes from the IIS7 cluster and test for availability and any delays. The goal is to have zero downtime.
- Pull a node from the SQL cluster and test for database availability. The goal is to have zero downtime.
Phase 1: Ensure that IIS 7 cluster nodes are syncing properly

Load the default IIS 7 website in IIS-Primary
We set the host priorities as #1 for the primary and #2 on the secondary. We then loaded the default IIS 7 webpage onto IIS-Primary and waited for the website to mirror onto IIS-Secondary. Within the IIS Manager window, we were able to determine that the Server Farm ‘cloudfarm’ had converged the webpages.

We then opened our web browsers and hit the IP address of IIS Primary and IIS Secondary to discover that the webpage had been uploaded onto both IIS Cluster Nodes.
We then opened the webpages config file located on IIS-Primary and made a change to the HTML. We watched the IIS Manager while constantly hitting the refresh button in our web browsers to see if we received any errors. Within 30 seconds we was the IIS Manager provision the website to IIS-Secondary and we were able to see the altered webpage on IIS-Secondary.
without any error pages.
Alter Webpage on IIS-Secondary
We attempted this same process by altering the IIS-Secondary config file and then waiting for the pages to sync. However, the process the IIS Cluster follows in its mirroring is to replicate IIS-Primary (set at Primary Node in IIS Manager) onto the other nodes. Therefore, any and all changes made to the website must be changed in IIS-Primary. Any changes made to the webpages in any other node other than the Primary Node will be overwritten within the IIS Cluster’s sync process. We see this as a smart feature as it prevents unwanted changes from being saved. For example, if two people were working on different versions of the webpages simultaneously, then the two would conflict and overwrite each other. By forcing all changes to be made to IIS-Primary, only a single user can make changes at any given time.

Load DentistPackage website onto IIS-Primary
We then loaded the DentistPackage website onto IIS-Primary and waited for the IIS 7 Cluster to mirror the site onto IIS-Secondary. Within a few seconds, IIS-Secondary was brought offline and the mirroring began. When the IIS cluster receives HTML requests while any node is down, the Cluster diverts all traffic to whichever servers are still online. This causes a delay in
receiving the updated webpage. Some users will receive an older version of the webpage that is served from a different cluster node. Once the server is updated with the webpage and brought back online, web traffic is then once again directed back to that specific node as well as all other nodes online. The IIS 7 Cluster does not redirect ALL web traffic to this updated node, but simply places the node back in the pool of ‘Servers Ready for Load’. Only web requests that are directed from the cluster to this specific node will receive the updated webpage until additional nodes have been brought ‘offline’, updated, and then placed back into the “Servers Ready for Load” pool. This causes some users to receive the updated webpage while others receive the non-updated page. There were no outages of web services while the nodes were brought offline and updated, even though there were delays in receiving the updated web content. The delays we notices were short and almost unnoticeably unless there was a major overhaul to the webpages, such as loading the DentistPackage over the Default IIS 7 page.

The DentistPackage website was updated and began to be served as the default webpage on our IIS-7 cluster. If we hit any specific node on the cluster or if we hit the cluster IP, we receive the new DentistPackage cluster.

Phase 2: Ensure that SQL Server 2008 cluster nodes are syncing properly

Loading DentistPackage Database on SQL-1 & SQL-2
The DentistPackage database was loaded on SQL-1. We had two team members log into each of the SQL server clusters and watch for updates as we made them. The DentistPackage database was copied over to SQL-2 without any issues.
In the Server Manager window under Features>Nodes, all nodes in the SQL cluster are listed and the status of each server is given.

Making Changes to the SQL Databases on each node

We created a SQL database user of TDRuser on SQL-1 and it synced over to SQL-2 immediately. We also made changes to some of the settings in SQL 1 and it mirrored over to SQL-2 immediately as well.
We did notice that with the Server Manager>Features>Nodes screen that SQL-2 would quickly go ‘down’ while it was updated.
Once the database was updated, the server returned to ‘Up’ status.

We attempted to make a change to the database on SQL-2 to see if it would be overwritten or if it would update the data on SQL-1. We updated a single line of data within the SQL-2 database and then looked for the change on SQL-1. By the time we found the line of data it was updated to match SQL-2’s change. Therefore, unlike the IIS 7 cluster, changes can be made to either SQL cluster node and the change will cascade to all subsequent nodes.

Phase 3: Testing redundancy, outages, and uptime
In this phase, we accessed the webpage by targeting the IIS cluster IP address of 206.71.69.170. We then removed IIS 7 Cluster nodes in an attempt to detect any errors, delays, or failures.

Test 1: Removing IIS 7 Cluster Nodes
We first verified that both IIS 7 Cluster nodes were online and ready for load. We also checked the status of our web page to make sure that everything was healthy before we removed the nodes.
Everything was healthy at this point!
We targeted the cluster IP address, and then we turned off the VM for IIS-Secondary, effectively pulling the power cable on IIS-Secondary. This in turn removed IIS-Secondary from the ‘cloudfarm’ IIS 7 cluster. We noticed NO CHANGE in the website or its services. The web content was delivered quickly.
The website ran flawlessly without a single glitch while IIS-Secondary was offline. We had several users, mainly our group, all access the website simultaneously and each of us was served a copy of the website without any errors.
When we then attempted to restart the VM for IIS-Secondary, the VM came back online but the IIS 7 ‘cloudfarm’ did not re-add the VM back to ‘Server Ready for Load’ pool. We had to manually go into IIS-1, the cluster NLB, and select ‘Repair Server’ for IIS-Secondary. Within 30 seconds the server was repaired and added back into the IIS cluster.
We then followed the identical procedure to remove IIS-Primary from the ‘cloudfarm’ cluster.
We had almost identical results with the IIS-Primary being brought offline while IIS-Secondary carried the load of the web traffic.
When we rebooted the VM IIS-Primary, we again had to select ‘Repair Server’ from the ‘cloudfarm’ manager within IIS-1. Once the IIS Manager repaired the server it was added back into the pool of servers ready for load.

**Test2: Removing SQL Cluster Nodes**
Following the same test methods as the IIS cluster nodes above, we removed SQL-1 from the cluster to uncover any errors and to test how the cluster rebuilt itself after a node lost power or connectivity or problems with the SQL cluster mirroring data between servers. We then brought down SQL-1 similar to how we brought down IIS-Primary.
We then powered on SQL-1 and waited for it to rejoin the SQL cluster automatically. It did without any errors or without having to be ‘Repaired’ like the IIS cluster nodes.
We then powered off SQL-2 following the same pattern as our test of SQL-1. We wanted to make sure that the cluster would remain ‘Up’ and would also re-add SQL-2 automatically once it was back online.
Once SQL-2 was rebooted and came back online, the SQL cluster once again added SQL-2 back to the cluster without having to be prompted and without ‘Repairing’ the server.
Test 3: Adding Non-configured Servers to the IIS Cluster Farm

We wanted to see what would happen if we took a non-configured server and added it to the IIS 7 cluster. We were hoping the cluster would configure the new node to be an IIS Server, replicate the settings, and then cascade the web site onto the new node. Essentially, we were testing to see if the IIS 7 Cluster would do all of the work once we had the cluster created.

We took SQL-4, a server running Windows Server 2008 with no additional roles, features, or applications. We originally created SQL-4 as a backup of SQL-1, SQL-2, and SQL-3 if anything were to go wrong with them. SQL-4 merely had all of its Windows updates and was a member of the domain cloud.local.

We accessed IIS-1, the NLB for the cluster, and selected “Add Server” on the right hand column.
We then enter the name SQL-4 into the dialog box and searched for it on the domain.
Once SQL-4 was located, we were able to select it from the available servers and click ‘OK’. We were hoping the server would then be added to the IIS-7 cluster and the replication of the configurations files and website should proceed.
We were wrong. The IIS -7 Cluster cannot add a new server to the cluster that does not have IIS Server added as a role previously. The cluster cannot replicate settings, configurations files, and application services. The cluster only mirrors the website files.
Sadly, the cluster is no longer available in IIS Manager and the status of our other server nodes has disappeared. We were unable to perform any action on the IIS services without receiving the notification alert that “Server ‘SQL-4’ does not exist in WebFarm ‘cloudfarm’”. Interestingly, we are still able to request the dentispackage website and receive it without delay or errors.
We then went to IIS-1 and tried to repair the server farm. The servers would not respond to the repair. We then rebooted IIS-Primary and IIS-Secondary. This did not fix the problem. The only way to fix the problem was to execute a revert on IIS-1, the NLB for the cluster. This restores the cloudfarm to its original state before we attempted to add SQL-4 to the cluster.
Lesson learned, DO NOT add anon-configured server node to the IIS-7 Cluster in hope that the cluster will configure, replicate, and attach the new node.

Each node needs the role of IIS Server added to it before it can be accepted into the IIS Cluster and made ready for load sharing.

Conclusion of Tests
Overall we feel that our tests were a great success. We discovered several key principals in making the cloud farm function properly. A few basic points include:

IIS Cluster Farm Principles
- Make all changes to IIS-Primary (or designated ‘Primary’ node)
- Any changes made to IIS-Secondary will be erased and overwritten and the node will inherit the properties of IIS-Primary
- While a node is brought offline for updating, all other nodes support the network traffic
- A single node can be brought offline with no delays in web services.
• In the middle of the Node Updating Process, some users will receive non-updated web pages stored on non-updated IIS nodes, while other users will receive the updated web page served from the updated IIS nodes.
• IIS cluster nodes need to be repaired once they are brought back online after being powered off.
• Each node needs the role of IIS Server added to it before it can be accepted into the IIS Cluster. If this is not done, the cluster breaks and is not repairable except through a revert of the Network Load Balancer (NLB).

SQL Cluster Farm Principles
• Changes can be made to either SQL-1 or SQL-2, or any other cluster node and the change will cascade to all remaining nodes.
• There is not SQL-Primary, all nodes can replicate to other nodes.

We feel that the IIS 7 cluster should not have to be manually ‘Repaired’ every time a node is physically brought offline. In the event of a power outage, the cluster should auto add the nodes once online back into the IIS cluster farm.

We also feel that the IIS Cluster should be able to take a basic non-configured server and replicate the settings, configuration files, and web pages to be identical to the rest of the other nodes. At minimum this attempt should not break the entire cluster forcing a restoration of each of the server nodes.

**Final Report/Recommendations**

The biggest change to the project we would advocate for anyone looking to duplicate this is to use physical machines wherever feasible. Having virtual machines concentrates the servers into one physical machine that could fail in any number of ways, including loss of power, hardware failure, or user error. Separating the machines into different physical boxes can take the pressure off of any one machine from a failure. If possible, spreading the machines physically over a greater geographical distance can help to decrease the chances of one failure knocking out multiple machines.

**IIS 7 Cluster**

The Microsoft IIS 7 Webfarm framework is currently in beta. As such it is not as robust as it could be, and many steps should be taken by prospective users to safeguard both the individual computers and the system as a whole. We learned throughout the project that trying to force the program to configure and function for us does not work. We recommend that anyone who wants to implement the IIS 7 Webfarm Cluster reads the documentation carefully to make sure that the servers are properly configured before they are added to the Webfarm. Additionally, after a
failure in the cluster, IIS 7 Webfarm framework cannot automatically repair and re-add the failed servers, the cluster must be manually repaired.

SQL 2008 Failover Cluster

When considering SQL clustering solutions, consider all the options, as well as their various strong and weak points. When using only two machines, a 100% uptime solution is impossible. Whether using a mirroring or failover clustering solution, in the event of a machine failure, there will be seconds of downtime no matter what. In our failover solution, when SQL-1 fails, it takes a small amount of time for SQL-2 to take ownership of the drives, as well as to start the Failover Server, MSDTC and Clustering services.

This solution, Failover Clustering, is known as “No Share” clustering, meaning that the various machines that are set up to fail over do not share the drives in the iSCSI SAN. One machine at a time has full possession and ownership of the drives, reading and writing from them when accessed by the Webserver(s). SQL Server is set up so that more than one user cannot alter the same data at the same time, so having more than one machine access the data at once is pointless, as the first node will lock out any other nodes that would try to access the shared disks. That is where the “no share” designation comes from. Only one machine can access the disks at any time, and for convenience the primary node will keep possession of the nodes until it fails over and another node takes control.

As in the IIS 7 case, we found that SQL Server 2008 R2 and its accompanying Failover Server utilities are very picky, and need to be configured very carefully or many hours will be spent trying to fix the problems that will unavoidably crop up.
Physical and Virtual Servers, and Roles

IIS-7 Cluster

IIS-Primary
(10.10.10.59)
(206.71.69.178)
Windows Server 2008 Enterprise
IIS-7 Cluster Node 1

IIS-Secondary
(10.10.10.54)
(206.71.69.179)
Windows Server 2008 Enterprise
IIS-7 Cluster Node 2

ZION Server
(10.10.10.170)

SQL-1
(10.10.10.58)
(206.71.69.181)
SQL Server 2008

SQL-2
(10.10.10.56)
(206.71.69.182)
SQL Server 2008

SQL-4
(10.10.10.57)
(206.71.69.183)
Windows Server 2008 Standard
(I don’t think this does ANYTHING?)

DC-1
(10.10.10.51)
Windows Server 2008 Standard
Domain Controller and DNS Server
Email 1:

Hi Tyson,

When your cluster group containing your Instance fails over to the other node, please be aware that the SQL instance is effectively shutting down and restarting on this 2nd node. This means that whilst it is doing so your SQL instance will be unavailable for around 30 seconds plus recovery time of your databases (in total usually around 40 seconds max). Therefore your web application will need to accommodate this situation and retry after brief timeout on connection failure or gracefully fail -for instance provide warning to the user about unavailability etc and would they like to retry ...you know the sort of thing :)

For troubleshooting purposes you should determine if an external machine to the cluster can connect to the SQL instance when it is running on either node. If so then you know that there is a configuration issue for your web server connection string.

For instance one possible problem you could be facing is with the port number that is assigned to the sql instance to listen on. If you have not configured a static port for your sql instance on the cluster, whenever it fails over it will be assigned a different port (since dynamic ports are sql server's named instance default behaviour). Have you hard coded a port number to your connection string for your web app? If so you should think about removing it..

Hope this helps and good luck!

btw, if on failure of one node you get automatic failover to the second node then it sounds to me like the cluster is working! You issue is simply one of configuration somewhere.

Email 2:

...should also have added that if you need some follow up with me, try and capture any exception messages from your web app if you can. Try and provide any detail possible in terms of connectivity and also dont forget to try/ check the obvious things i.e. Use ping to check resolution, check server firewalls, dns config, connection string etc etc.

Email 3:

Hi Steve,

No one has come back with any questions so far.
Yeah you are right, the cluster wont be available 100% of the time since there is the possibility of failover, but then no solution can be guaranteed to be 100% available -ever. Windows and
SQL Clustering is known as "shared nothing" which means just that, nothing is shared. There will only ever be 1 owner of a cluster group at any one time. The sort of thing I think you are thinking about is Oracle's implementation call RAC (real application clusters) which is a shared everything and is just a different approach and has its own drawbacks too. SQL Server can do read only shared db (see http://tenbulls.co.uk/2011/05/25/sqlrally-scalable-shared-database-demo-redux/).

You can tier HA solutions too, and SQL Server does provide lots of different possibilities. The one which is probably most likely for you is Database Mirroring with Automatic fail-over http://technet.microsoft.com/en-us/library/ms175191.aspx

In your connection string you can specify a fail-over partner which once the db has failed over will autoredirect the application connection see http://msdn.microsoft.com/en-us/library/5h52hef8%28v=vs.80%29.aspx

Mirroring failover can be very very quick in comparison to clustering however each technology has its benefits and drawbacks. In brief you should use Mirroring for providing HA to your db and Clustering to provide HA to your instance. If you just went for mirroring then you would have to ensure that you manually synchronise your instance level components (if they change) such as the logins or agent jobs etc. Mirroring is also an SQL Enterprise Edition feature sadly.

Other technologies you can also look at for HA is Peer to Peer replication (might be a good one for your load balancing suggestion) http://msdn.microsoft.com/en-us/library/ms151196.aspx or Service Broker.

Hope this all made sense :)

btw Dont know if you got it or not but there's a vid of the presentation if its useful http://sqlbits.com/Sessions/Event8/SQL_Server_Clustering_for_Dummies