

# Building a Scale-Out SQL Server 2008 Reporting Services Farm

---

**Edwin Sarmiento**

This white paper discusses the steps to configure a scale-out SQL Server 2008 R2 Reporting Services farm environment running on Windows Server 2008 R2 network load balancer.

# Table of Contents

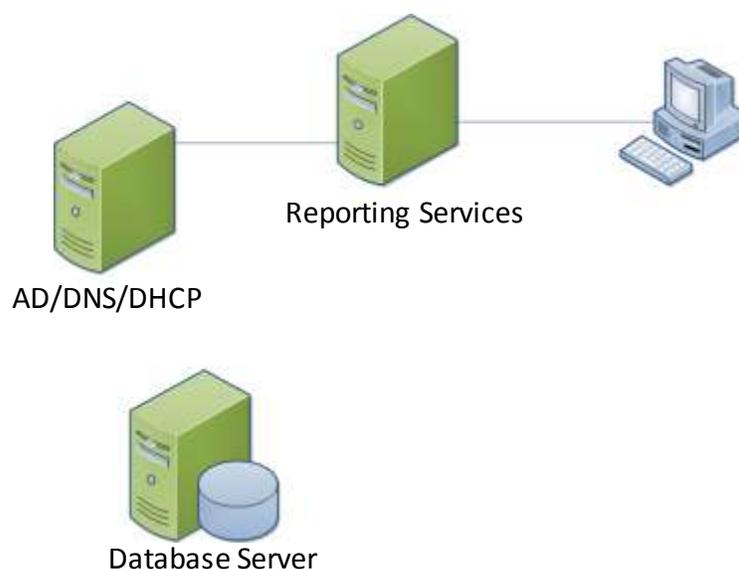
Network Infrastructure.....	2
What is network load balancing? .....	3
Preparing your network.....	4
Create a DNS entry for the NLB cluster application .....	4
Configure the server network cards.....	5
Adding the Network Load Balancing Feature .....	7
Creating the NLB Cluster .....	9
Adding hosts to the NLB cluster .....	18
Installing SQL Server 2008 R2 Reporting Services on the NLB Cluster .....	21
Configuring the first SQL Server 2008 R2 Reporting Services instance .....	29
Configuring the second SQL Server 2008 R2 Reporting Services instance .....	35
Joining the second SQL Server 2008 R2 Reporting Services instance to the Scale-Out Deployment .....	40
Configuring view state validation .....	41
Configuring the Hostname and UrlRoot.....	44
Workarounds for the HTTP 401 Error Message.....	45
About The Author .....	47

## Building a Scale-Out SQL Server 2008 R2 Reporting Services Farm

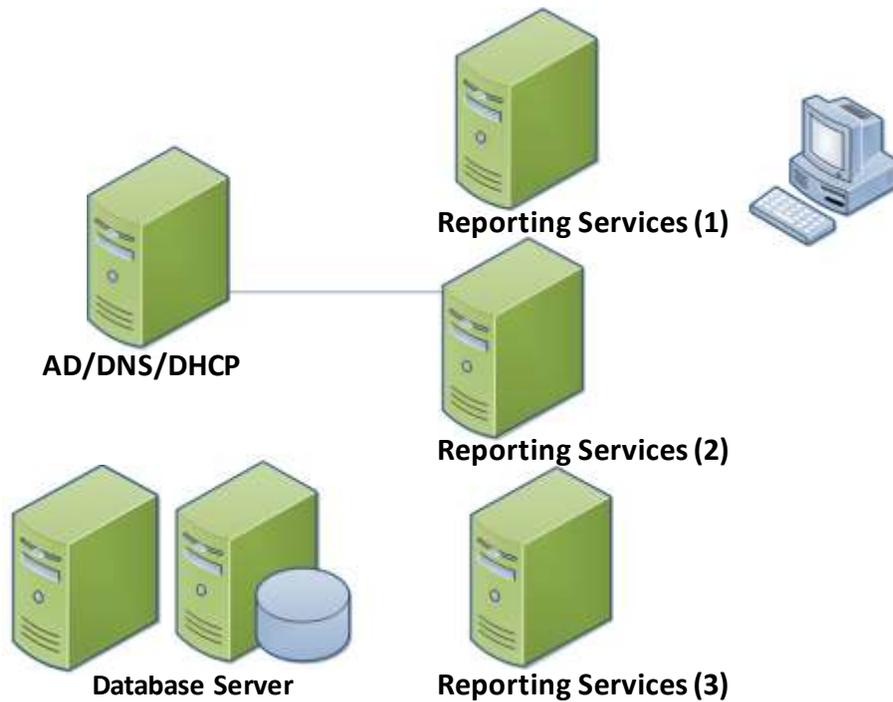
In a SQL Server 2008 R2 Reporting Services scale-out deployment, two or more report server instances share a single report server database. This type of deployment enables you to increase the number of users who concurrently access reports and improve the availability of the report server.

### Network Infrastructure

Most Reporting Services deployments are either based on a single-server deployment or, at most, a two-server deployment where the Reporting Services instance is running on a different server from the database server. A diagram of the two-server deployment is shown below.



This type of deployment is limited not just because of the capacity of the servers to handle additional workload but also in providing fault-tolerance to the entire Reporting Services infrastructure. Scaling out the deployment to multiple servers not only provides additional resources to handle increase in workload but also fault-tolerance and high availability in the event that one or more of the components fail. A sample diagram of the scale-out Reporting Services instance deployment is shown below. Note that the database server is configured to run on a failover cluster for high availability.



## What is network load balancing?

Network load balancing (NLB) is a clustering technology offered by Microsoft as part of the Windows Server operating systems. The good thing about Windows NLB is that it's available on nearly all the editions of Windows Server 2008 R2, unlike the failover clustering feature where you need at least the Enterprise Edition to have it configured. NLB uses a distributed algorithm to provide network load-balanced traffic for IP-based services such as web, virtual private networking, streaming media, terminal services, proxy, and so forth. This makes Windows NLB an ideal choice for SQL Server 2008 R2 Reporting Services because it's hosted as a web service.

Similar to the failover clustering technology, NLB assigns a virtual IP address to the cluster. When a client request is made using this virtual IP address, the NLB maps it to the physical IP address of one of the cluster nodes based on the configurations made as well as availability of the nodes. As far as the end user is concerned, only a single server is serving the request. The NLB nodes run separate copies of the server application—in this case, SQL Server Reporting Services. Because Reporting Services requires a backend SQL Server database for the report metadata, the NLB nodes should be able to access a shared database in order for the data access and updates to be synchronized. A supported scenario is to have the report server database on its own server where it can be accessed by all nodes of the NLB cluster.

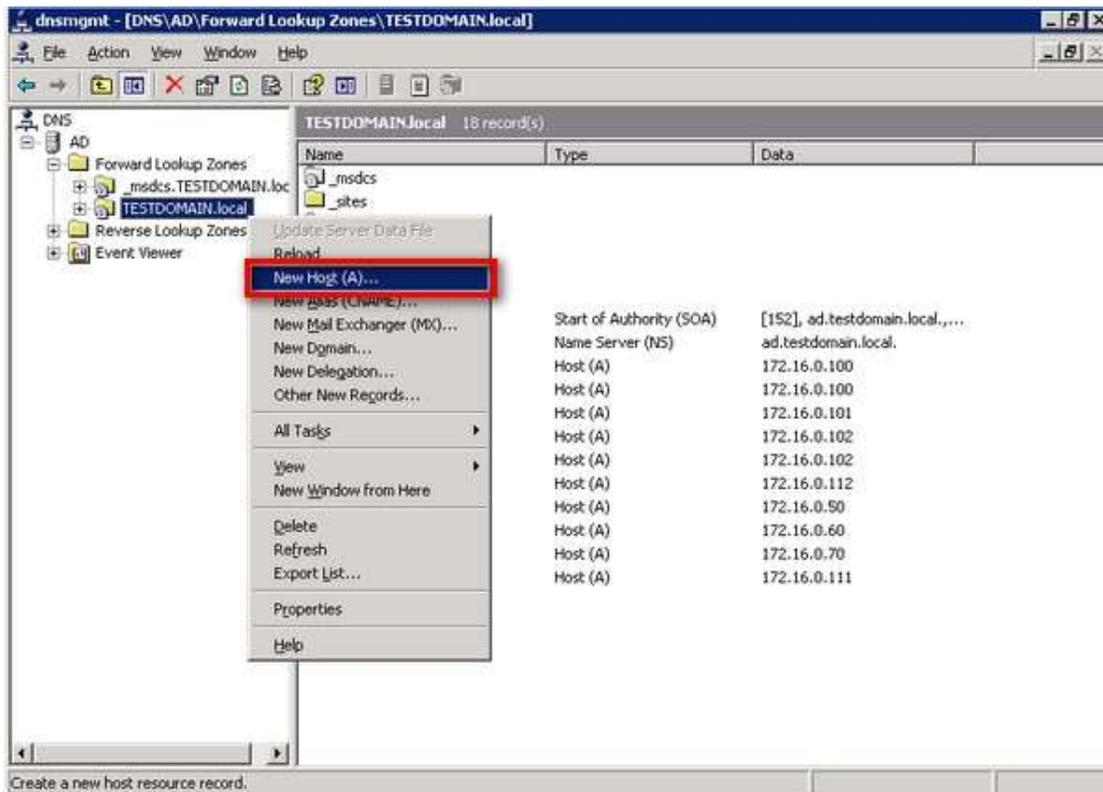
## Preparing your network

Although the task of preparing your network isn't typically a DBA's responsibility, it's important to understand what needs to be prepared prior to deploying your Windows NLB cluster. This is one of the reasons why you need to be on good terms with your systems administrator.

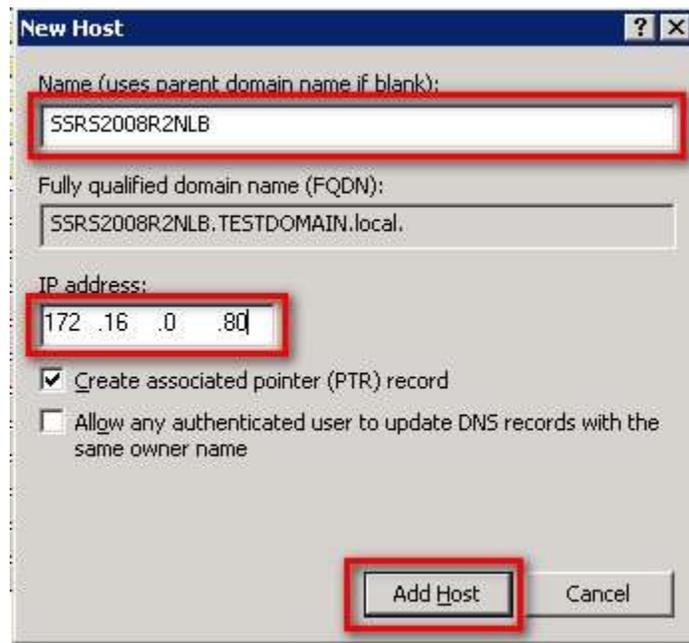
### Create a DNS entry for the NLB cluster application

Your DNS administrator will have to create a DNS entry for the NLB cluster application unless you're granted permissions to administer your DNS servers. Users will access the NLB cluster using a friendly name, so you must create a DNS entry mapping to the virtual IP address that will be used by the NLB cluster for SQL Server 2008 R2 Reporting Services. Make sure that you already have an address allocated for the virtual IP.

To create a DNS entry for the NLB cluster application, open DNS Administrator from the Microsoft DNS server. Right-click on the domain name and select **New Host (A)...**



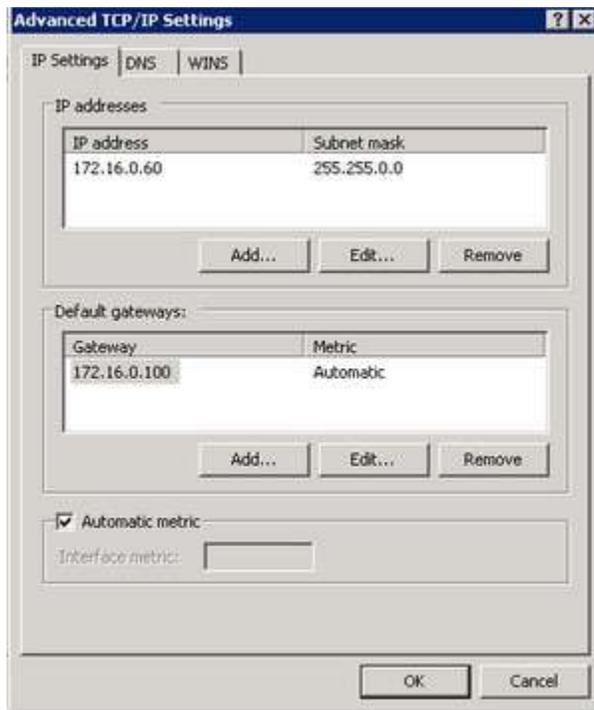
On the **New Host** property page, enter the NLB cluster application name as well as the virtual IP address. Click **Add Host**



For this example, use the name **SSRS2008R2NLB** with an IP address of **172.16.0.80**. This means that requests made to the application name will be directed to the specified IP address. You can have multiple applications hosted in the NLB cluster, so it's important to have different application names with their corresponding virtual IP address. For this example, you'll host only one application: the SQL Server 2008 R2 Reporting Services instance.

## Configure the server network cards

Depending on how your network infrastructure is configured, you may have different network card configurations. Your servers may have only a single network card configured with a single IP address or multiple network cards each with its own IP address. Configuration of the NLB cluster will depend on how your network cards are configured. It's important to consider the IP address that you'll use to listen to requests — this will be the IP address that you need to use for the NLB cluster. Unlike in failover clustering technology, where you'd need an additional network card to act as the heartbeat, NLB can create a unique cluster adapter based on how your network adapter is configured. I recommend that you have an additional network adapter on a different subnet from the production network specifically for heartbeat use to improve communication between cluster nodes. For this example, I only have a single network card with a single IP address.

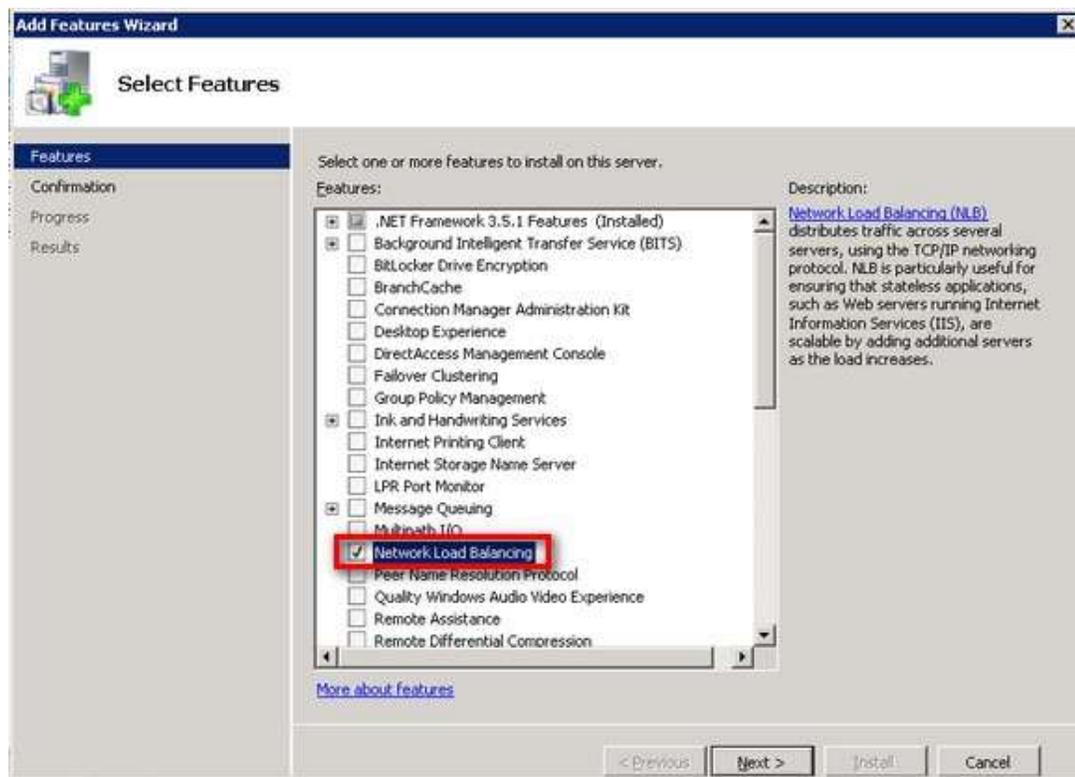


After the NLB cluster has been configured and the nodes added, we will check the network card configuration again to see what the NLB cluster configuration has done.

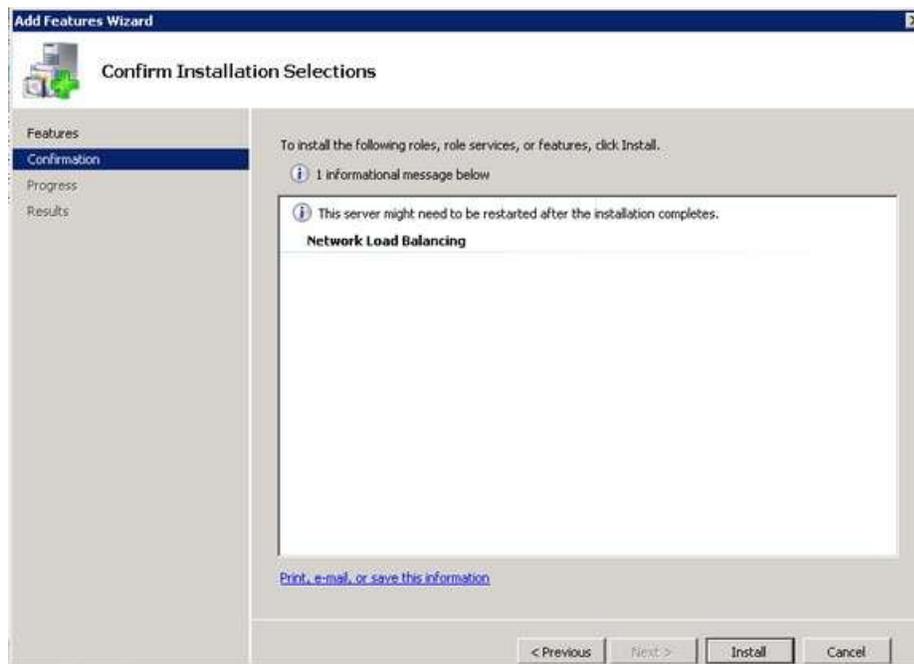
## Adding the Network Load Balancing Feature

Similar to enabling the failover clustering feature in Windows Server 2008 R2, we will use the **Server Manager** console to add the **Network Load Balancing** feature. To add the Network Load Balancing feature:

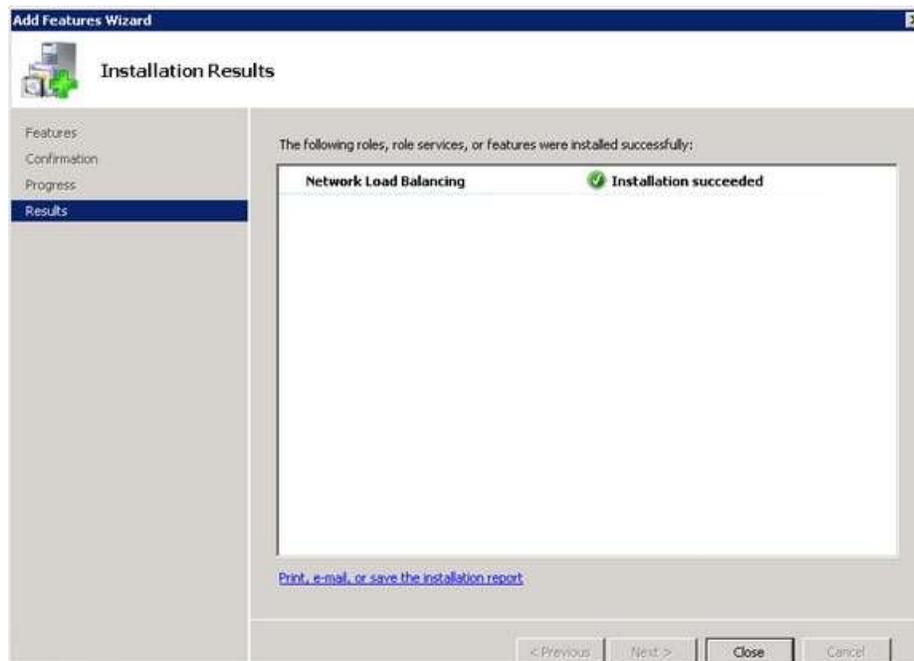
1. Open the **Server Manager** console and select **Features**.
2. Click the **Add Features** link. This will run the **Add Features Wizard**
3. In the **Select Features** dialog box, select the **Network Load Balancing** checkbox and click **Next**. *(And while you're at it, you can include the **.NET Framework 3.5.1 Features** as well as this will be used by the SQL Server 2008 R2 installation)*



4. In the **Confirm Installation Selections** dialog box, click **Install** to confirm the selection and proceed to do the installation of the Network Load Balancing feature



5. In the **Installation Results** dialog box, click **Close**. This completes the installation of the Network Load Balancing feature on the first node.

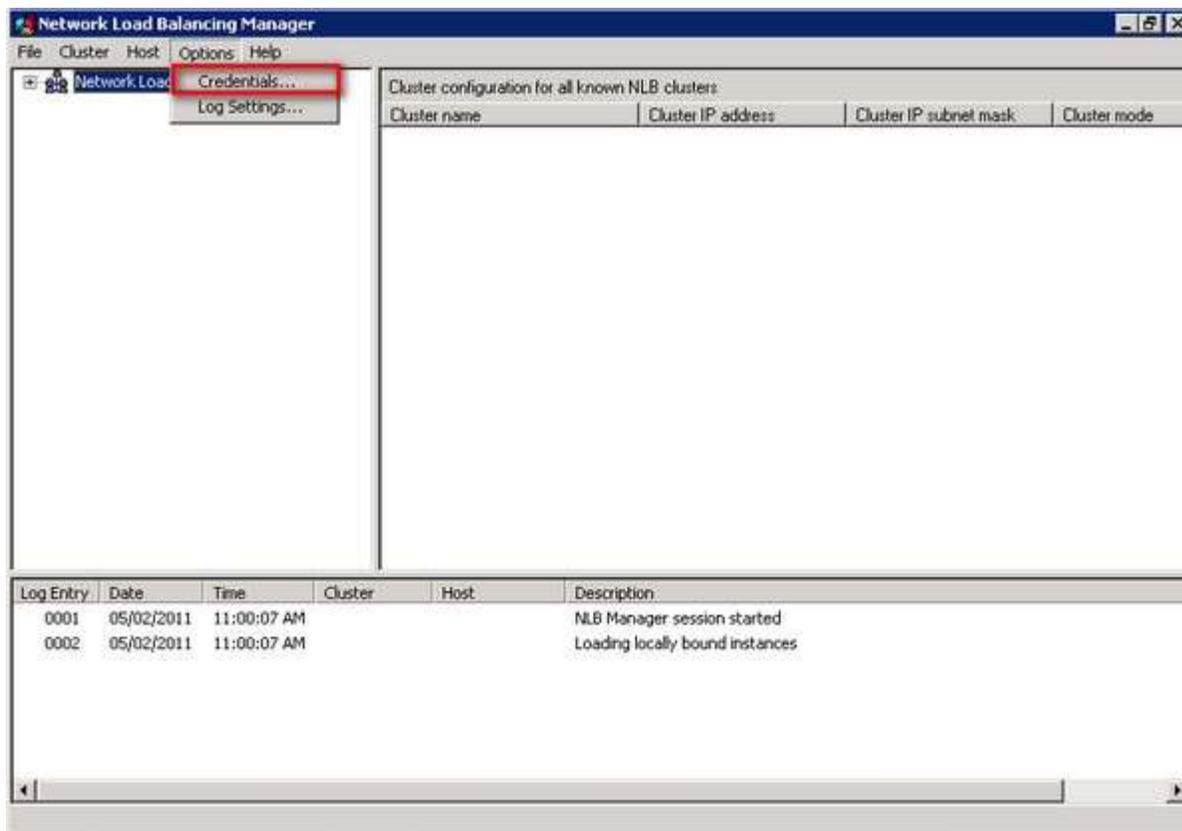


You must repeat these steps on all the other nodes of the NLB cluster before proceeding with creation of the new NLB cluster.

## Creating the NLB Cluster

As the Network Load Balancing feature has been installed on both of the servers, we can now create network load balanced clusters using the Network Load Balancing Manager console. You can run this administrative console on any of the servers that will become nodes of the NLB cluster. To create the NLB Cluster,

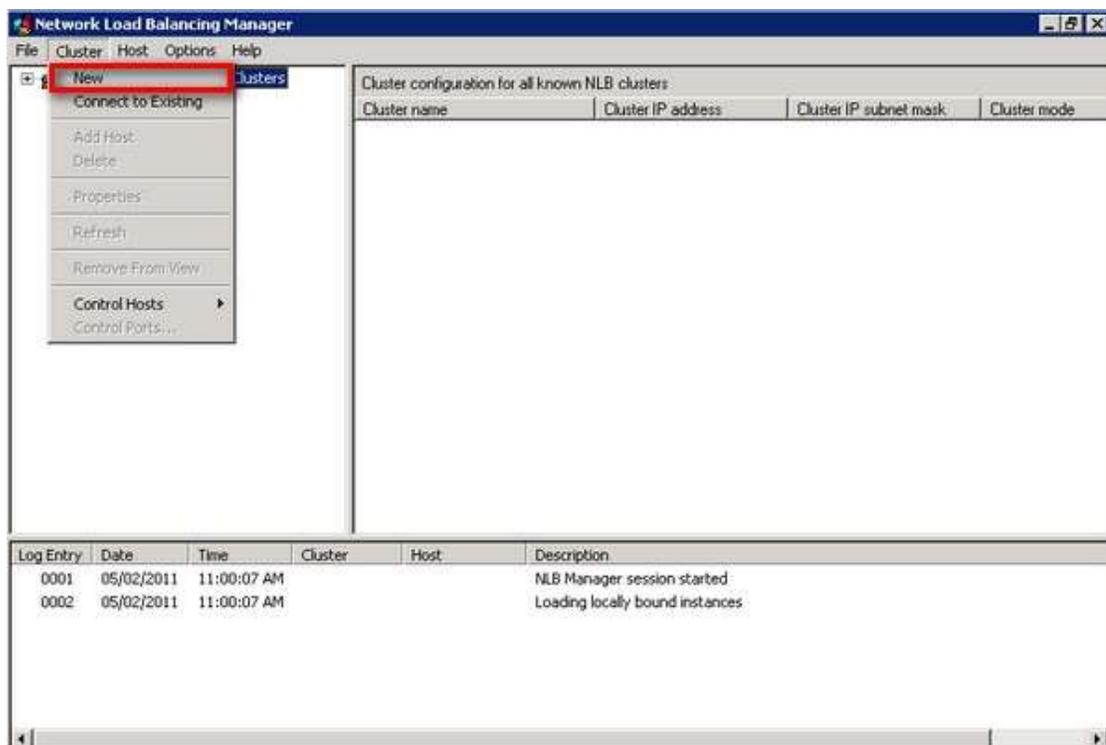
1. Open the **Network Load Balancing Manager** console. In order to create a new NLB cluster, your account needs to be a member of the local Administrators group of the servers that will be a part of the cluster. However, you can configure a different account that will be used to do the creation of the cluster. To do that, select the **Options** menu and click **Credentials..**



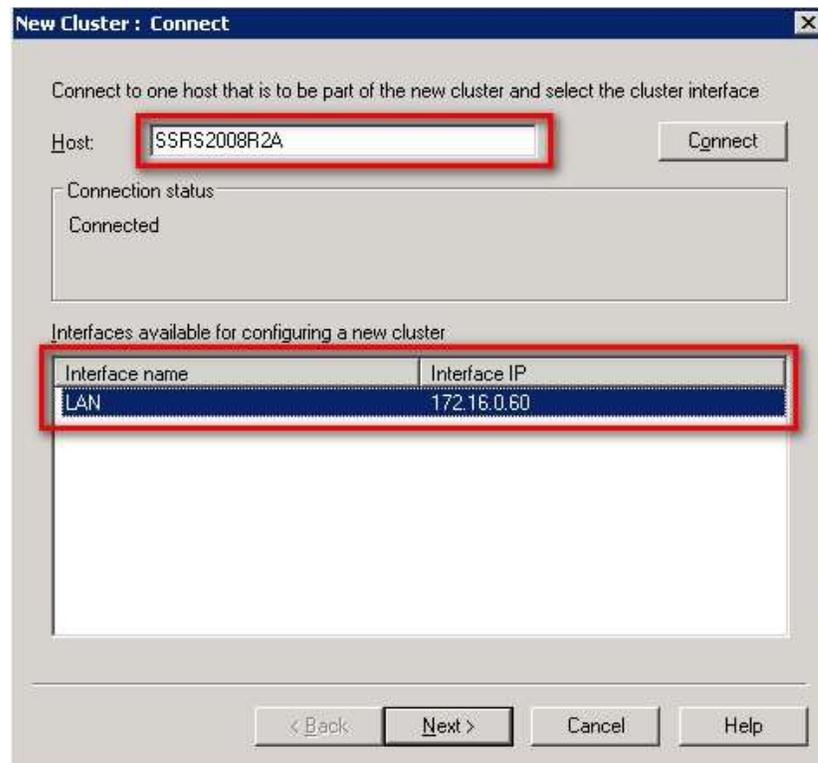
You will be prompted to enter the credentials of a login that will be used to connect to the servers as you are creating the NLB cluster.



2. Proceed with the creation of the NLB cluster. Under the **Cluster** menu, select **New** to create a new NLB cluster. This will run the **New Cluster** wizard



3. On the **New Cluster : Connect** page, enter the hostname or IP address of the first server that will be a part of the NLB cluster. Click **Connect**. Network Load Balancing Manager detects the available network cards on the server and uses a specific IP address that will be used by the NLB cluster traffic. The network card configuration will play an important role in how your NLB cluster works.



NLB clustering can be configured using different types of models, depending on the number of network cards available on the servers. While it is recommended to use more than one network card for optimum performance and high availability, a single network card configuration works just fine. The different types of NLB model are described below

- *Single network adapter in Unicast Mode* - this means that your servers only have a single network card. This configuration is suitable when network communication among cluster hosts is not required and in which there is limited dedicated traffic from outside the cluster subnet to specific cluster hosts. This also means that nodes in the cluster can communicate with addresses outside the cluster subnet, but communication between the nodes within the cluster subnet is not possible. Plus, the MAC address of the network card is disabled and the cluster MAC address is used.
- *Multiple network adapters in Unicast Mode* - as the name implies, you have multiple network cards in your servers which may or may not be dedicated for NLB networking such as backups, monitoring and management. This configuration is suitable when network communication among cluster hosts is required. It is also appropriate when you want to separate the traffic used to manage the cluster from the traffic occurring between the cluster and client computers. Nodes within the cluster can communicate with each other within the cluster subnet and also with addresses outside the subnet. As this configuration is still in unicast mode, the MAC address of the network card is disabled and the cluster MAC address is used.
- *Single network adapter in Multicast Mode* - again, single network card on the server but the NLB cluster is configured in multicast mode. This configuration is suitable when network communication among cluster hosts is required but there is limited dedicated traffic from outside the cluster subnet to specific cluster hosts. Nodes within the cluster

can communicate with each other within the cluster subnet and also with addresses outside the subnet. Unlike in unicast mode, both the network card and cluster MAC addresses are enabled.

- *Multiple network adapters in Multicast Mode* - this configuration is suitable for a cluster in which ordinary network communication among cluster hosts is necessary and in which there is heavy dedicated traffic from outside the cluster subnet to specific cluster hosts. Nodes within the cluster can communicate with each other within the cluster subnet and also with addresses outside the subnet and that both the network card and cluster MAC addresses are enabled.

Understanding how you want your NLB cluster to function will help guide you in configuring your network cards prior to creating the cluster. For this example, I will be using a *single network card configured in multicast mode* as I want the simplest configuration as well as be able to allow communication between my cluster nodes as well as within the network.

4. On the **New Cluster : Host Parameters** page, select a value in **Priority (Unique host identifier)**. This parameter specifies a unique ID for each host that you will add in your NLB cluster. 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. Think of it as the active node in the failover cluster. Click **Next** to continue.

**New Cluster : Host Parameters**

Priority (unique host identifier): [Dropdown menu]

Dedicated IP addresses

IP address	Subnet mask
172.16.0.60	255.255.0.0

Add... Edit... Remove

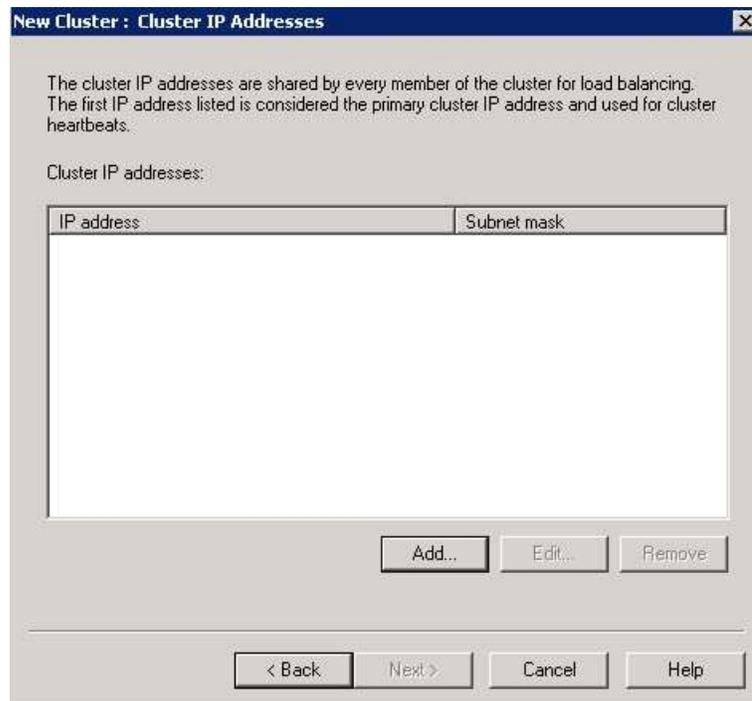
Initial host state:

Default state: [Started]

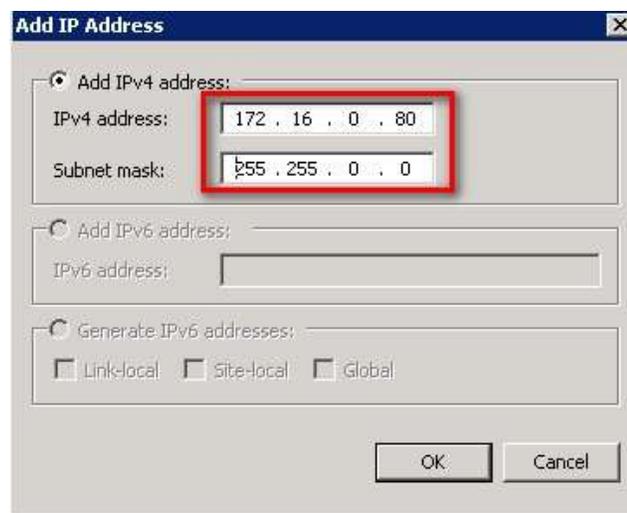
Retain suspended state after computer restarts

< Back Next > Cancel Help

5. On the **New Cluster : Cluster IP Addresses** page, click **Add** to enter the cluster IP address that is shared by every host in the cluster.

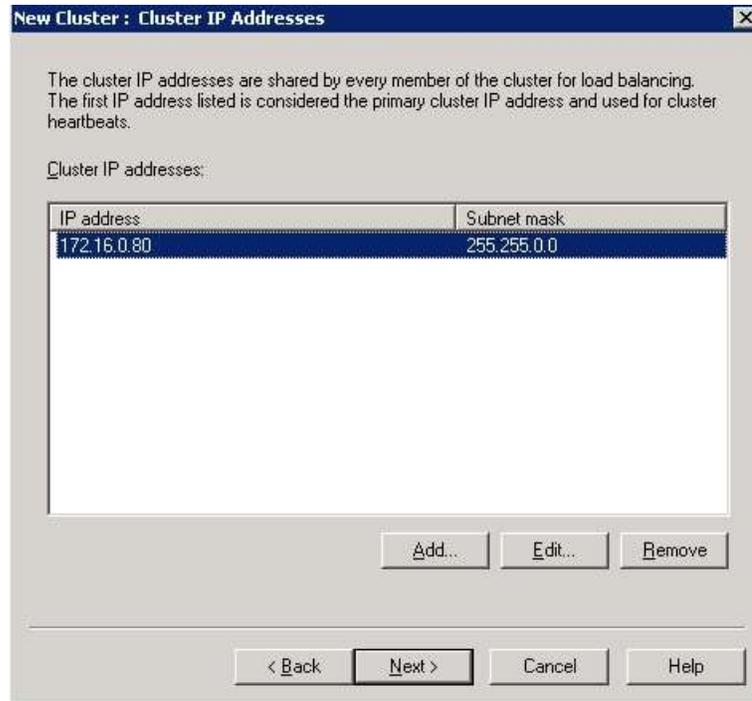


Enter the virtual IP address of the NLB cluster and click **OK**. This IP address will be used by the clients to communicate with the application - in our case, the SQL Server 2008 R2 Reporting Services instance. This IP address is the same as the one we added in our DNS server: **172.16.0.80**.



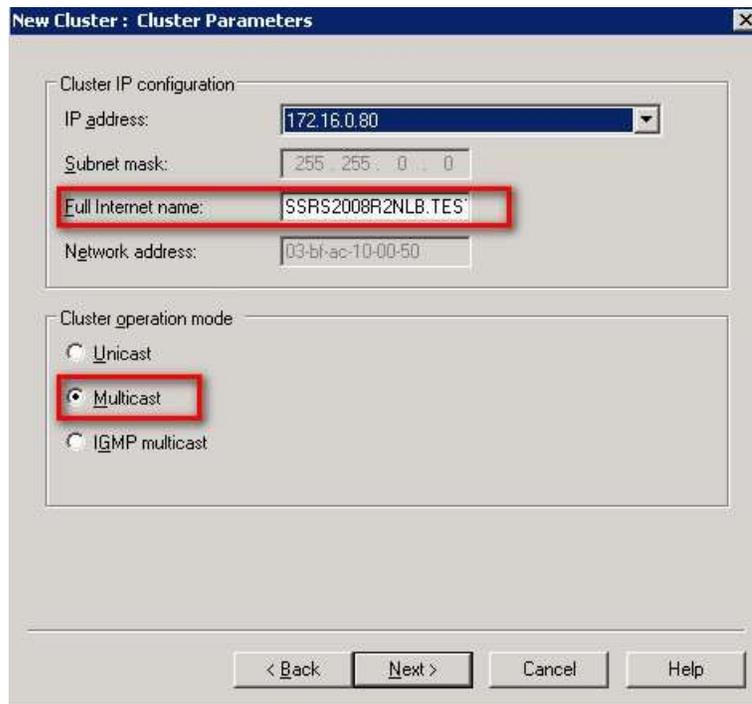
NLB adds this IP address to the TCP/IP stack on the selected network card of all hosts chosen to be part of the cluster. We will see this later when we check on the new TCP/IP configuration of the network card after the NLB cluster is created. NLB supports IPv6 but doesn't support Dynamic Host Configuration Protocol (DHCP). Since I don't have IPv6 configured on my network cards, the option is disabled. NLB also disables DHCP on each network card it configures, so the IP addresses must be static.

Note that the first IP address listed is considered the primary cluster IP address and used for cluster heartbeats for this configuration - single network adapter in multicast mode. This information will be useful later on when you want to capture network traffic as part of troubleshooting. Click **Next** to continue.

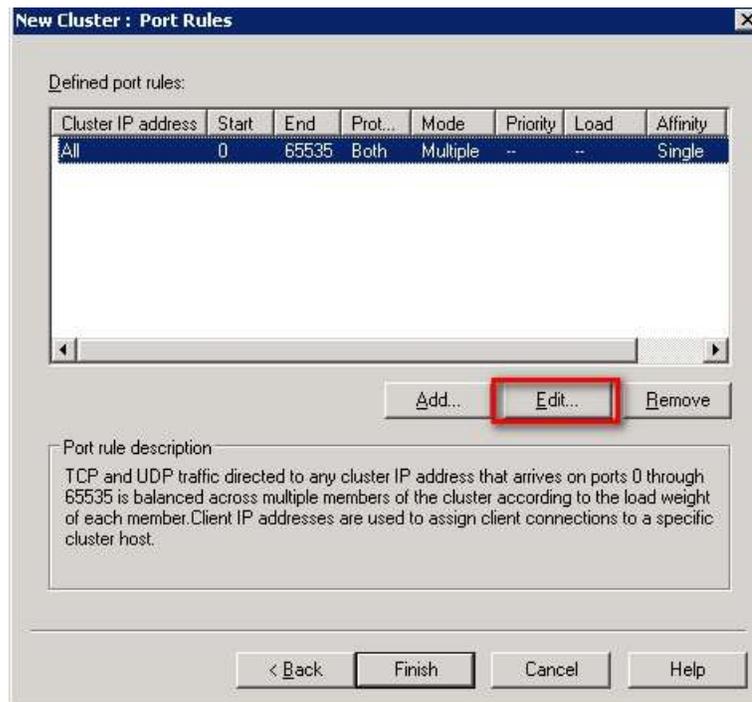


6. On the **New Cluster : Cluster Parameters** page, under the **Cluster IP configuration** section, enter the fully qualified domain name of the application that will be running on the NLB cluster. In this example, I use the DNS entry that we created earlier - **SSRS2008R2NLB.TESTDOMAIN.local**

In **Cluster operation mode** section, select **Multicast** and click **Next**.



7. On the **New Cluster : Port Rules** page, click the **Edit** button to modify the default port rules.



This will open the **Add/Edit Port Rule** page. Because SQL Server 2008 R2 Reporting Services is a web service, we will use the default port number for HTTP traffic - port 80. In cases where you would use a non-default port number for the web service, assign that port

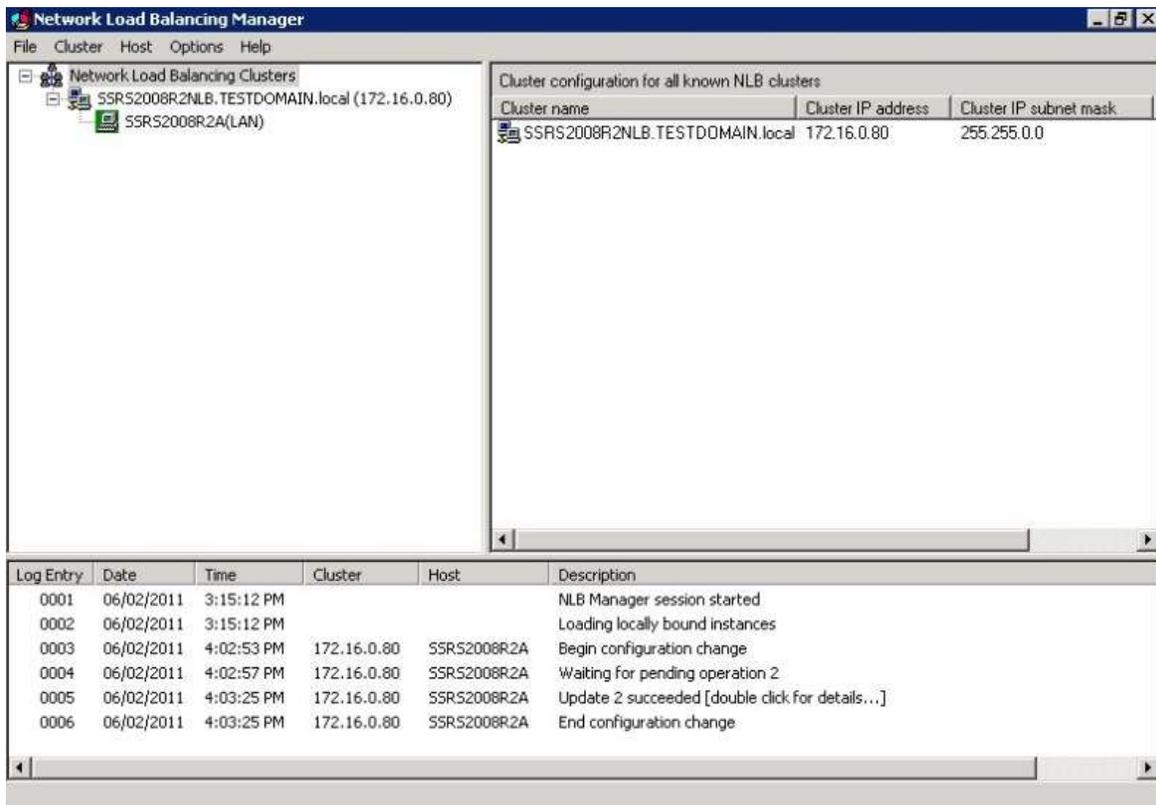
number in this page. We will use the **Port Range** section for this task and, for security reasons, limit the port number to only allow traffic for that specific port number. In the **Protocols** section, select **TCP**. In the **Filtering mode** section, select **Multiple host**. This specifies that multiple hosts in the NLB cluster will handle network traffic for this port rule. In the **Affinity** section, select **Single**. This configuration is commonly used for web servers especially when the application is not capable of maintaining state across servers. Single affinity forces clients to use the same server as the previous connection, thus, maintaining session state. While traffic from multiple clients will be distributed over multiple cluster nodes, the load may not be spread evenly, because, depending on the application requirement, one client could generate more traffic and sessions than another client. Click **OK** to close the **Add/Edit Port Rule** page.

The screenshot shows the 'Add/Edit Port Rule' dialog box with the following settings:

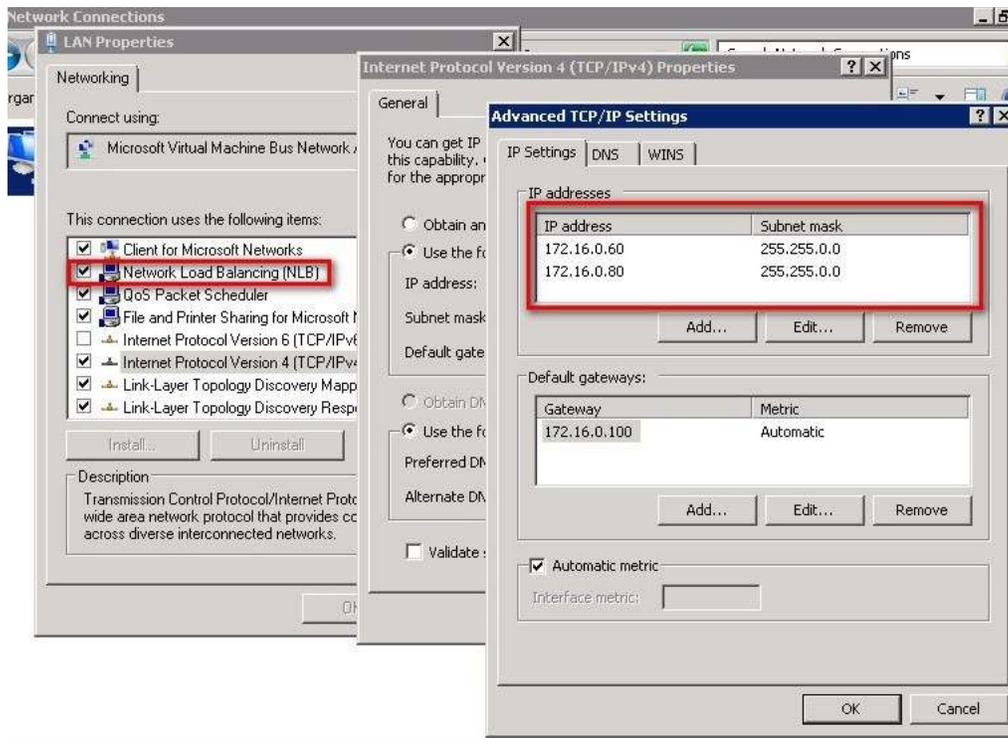
- Cluster IP address: [ ] or  All
- Port range: From: 80 To: 80
- Protocols:  TCP  UDP  Both
- Filtering mode:  Multiple host  Single host  Disable this port range
- Affinity:  None  Single  Network
- Timeout(in minutes):  0

8. On the **New Cluster : Port Rules** page, click **Finish** to create the NLB cluster.

This will create a new NLB cluster with just a single server.



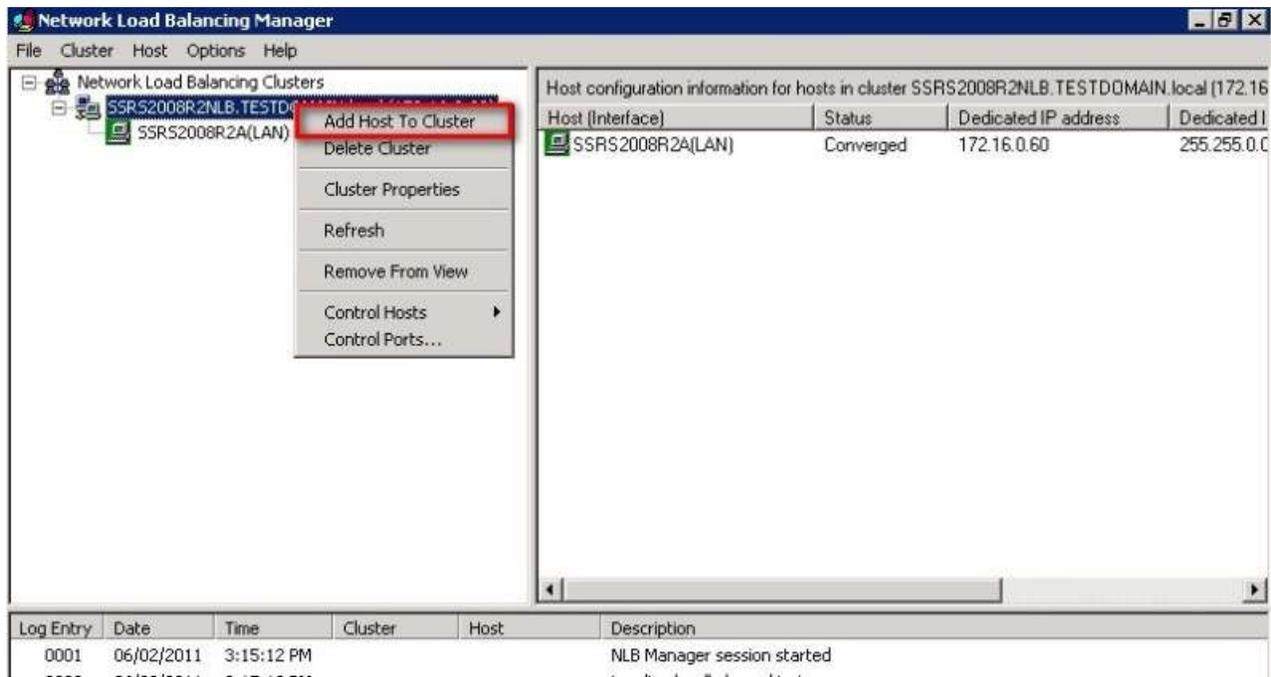
Let's take a look at the new TCP/IP configuration of the network card on the first node that we've added in the NLB cluster



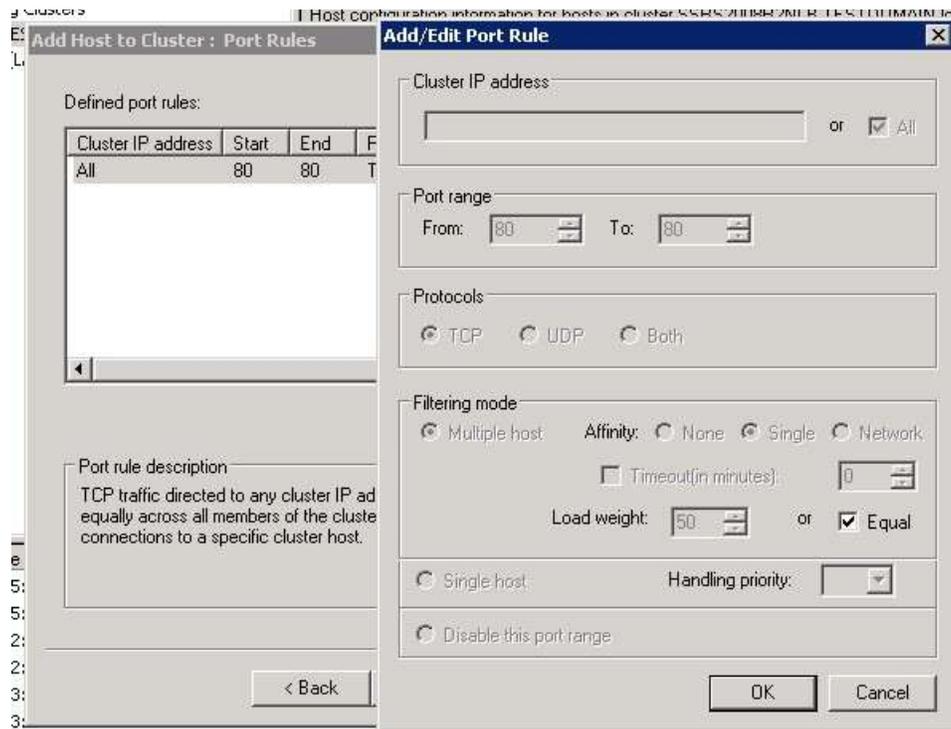
The configuration settings we did while creating the new NLB cluster is now implemented in the network card - Network Load Balancing (NLB) is already installed and enabled with the virtual IP address assigned for the NLB cluster is added in the list of IP addresses.

## Adding hosts to the NLB cluster

To add hosts to the cluster, right-click the new cluster, and then click **Add Host to Cluster**. Configure the host parameters (including host priority and dedicated IP addresses) for the additional hosts by following the same instructions that you used to configure the initial host.



You're adding hosts to an already configured cluster; which means all the cluster-wide parameters remain the same; therefore, these options, such as the **Add/Edit Port Rule** page, will be disabled.



Similar to how you would test a Windows failover cluster, you can run a continuous PING test on the virtual IP address or the fully qualified domain name while rebooting one node at a time to see

how the NLB cluster responds. Because this NLB cluster will host SQL Server 2008 R2 Reporting Services, a Telnet test on port 80 would be helpful to see if the cluster is indeed listening. That would be appropriate after installing Reporting Services on all cluster nodes.

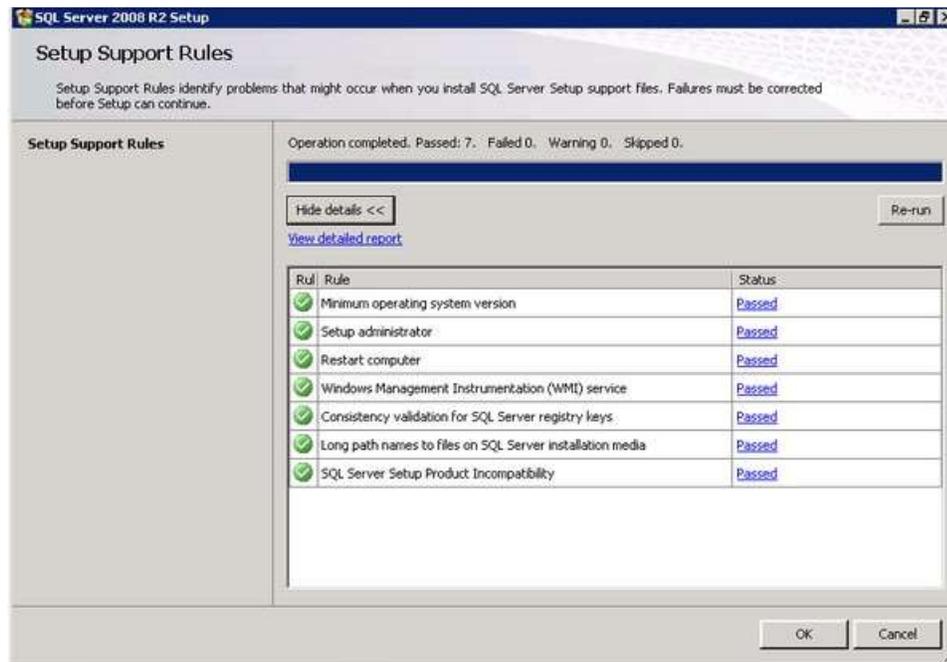
## Installing SQL Server 2008 R2 Reporting Services on the NLB Cluster

You're installing SQL Server 2008 R2 Reporting Services R2 on the NLB cluster, so you'll have to perform these steps on both of the nodes. Because the .NET Framework 3.5.1 is already installed as part of the steps outlined earlier, you can immediately proceed with running the **setup.exe** file:

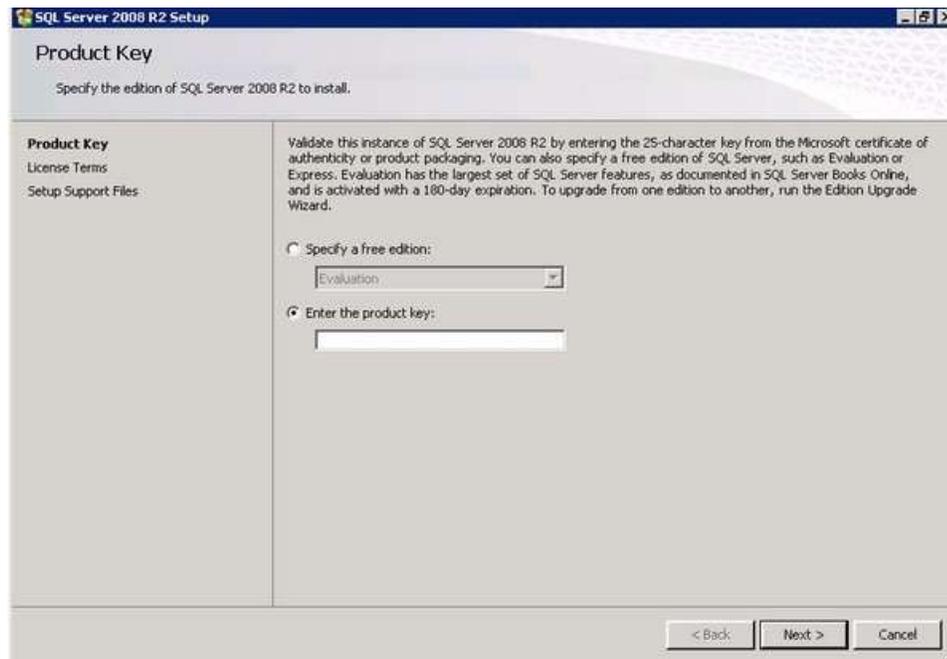
1. On the **SQL Server Installation Center**. Click on the **Installation** link on the left-hand side and click the **New installation or add features to an existing installation** link. The **SQL Server 2008 Setup** wizard launches.



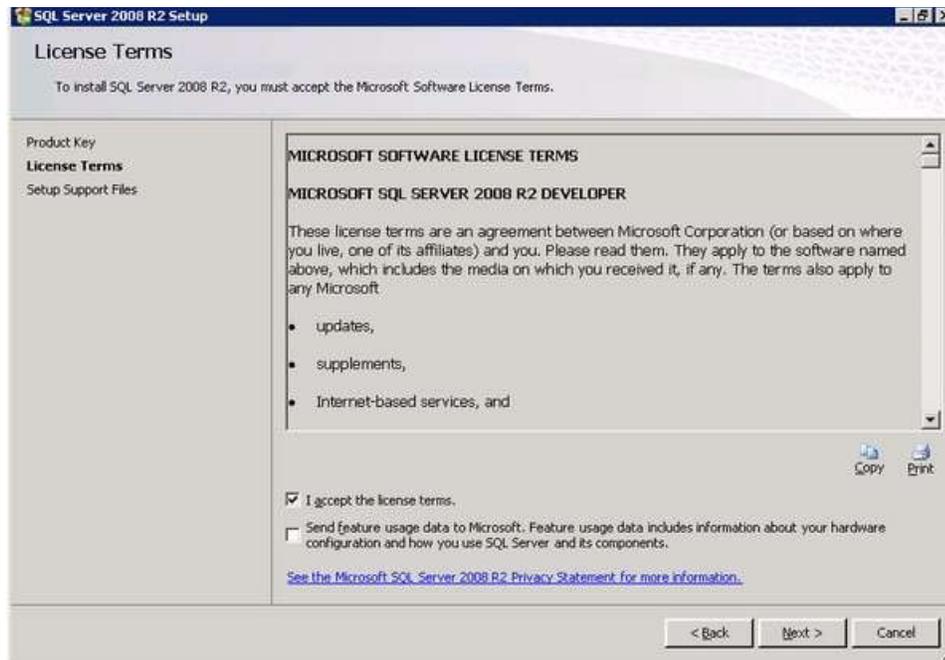
2. In the **Setup Support Rules** dialog box, validate that the checks return successful results and click **OK**.



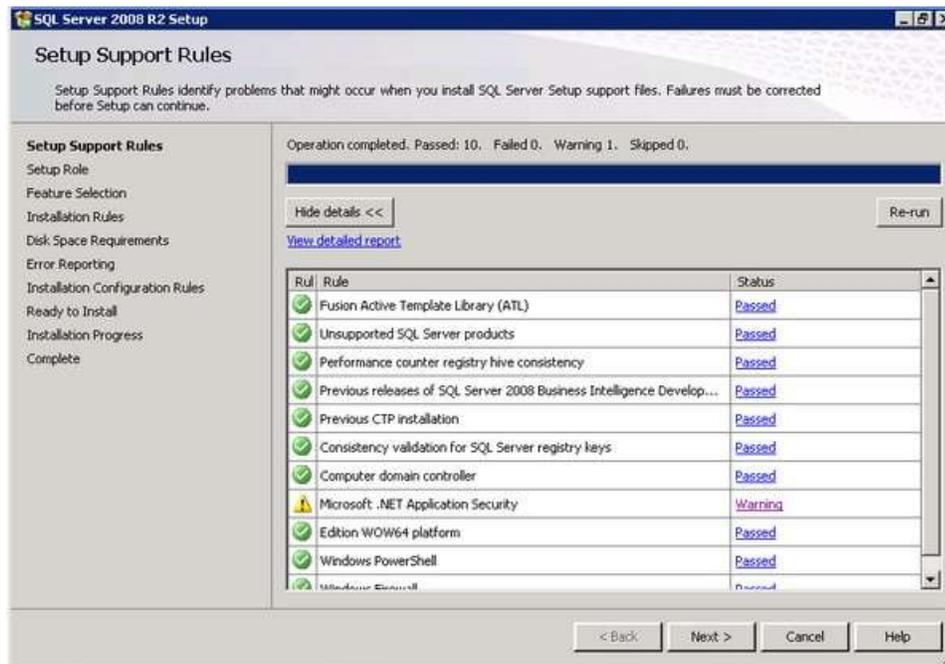
3. In the **Product Key** dialog box, enter the product key that came with your installation media and click **Next**.



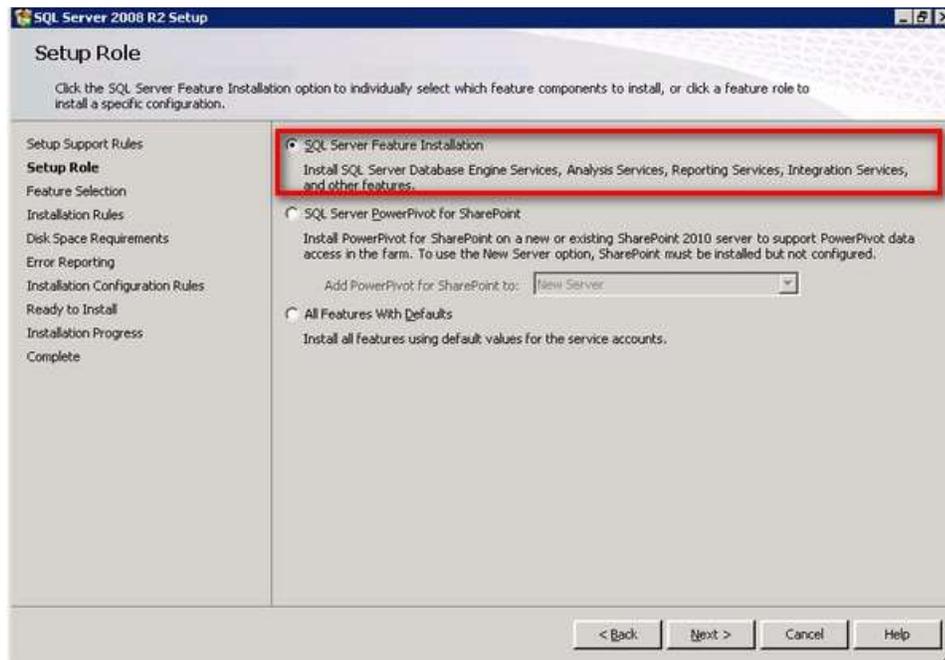
4. In the **License Terms** dialog box, click the **I accept the license terms** check box and click **Next**.



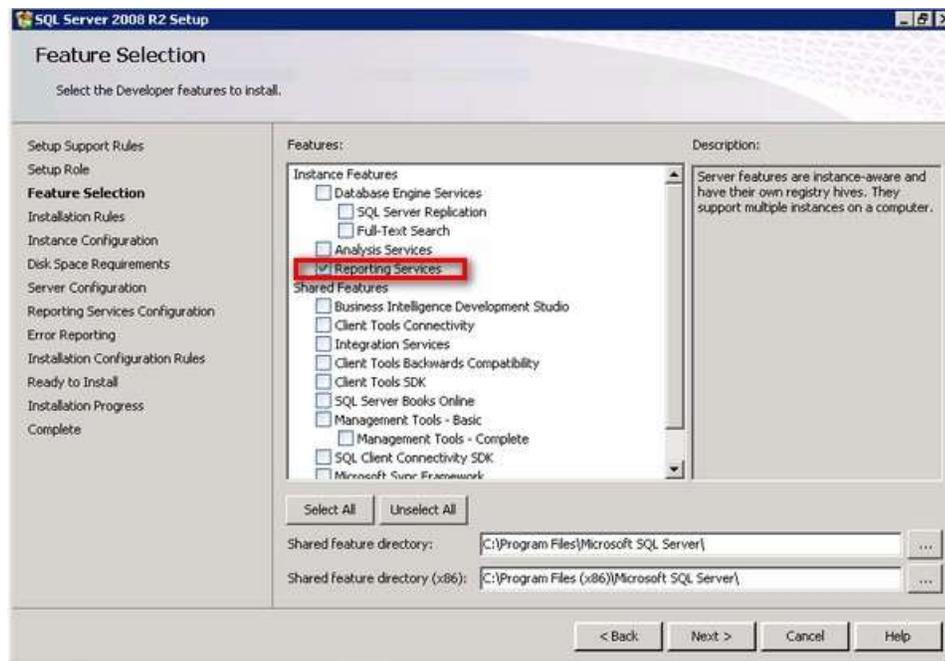
5. In the second **Setup Support Rules** dialog box, click **Install**. Validate that the checks return successful results. If the checks returned a few warnings, make sure you fix them before proceeding with the installation. Click **Next**.



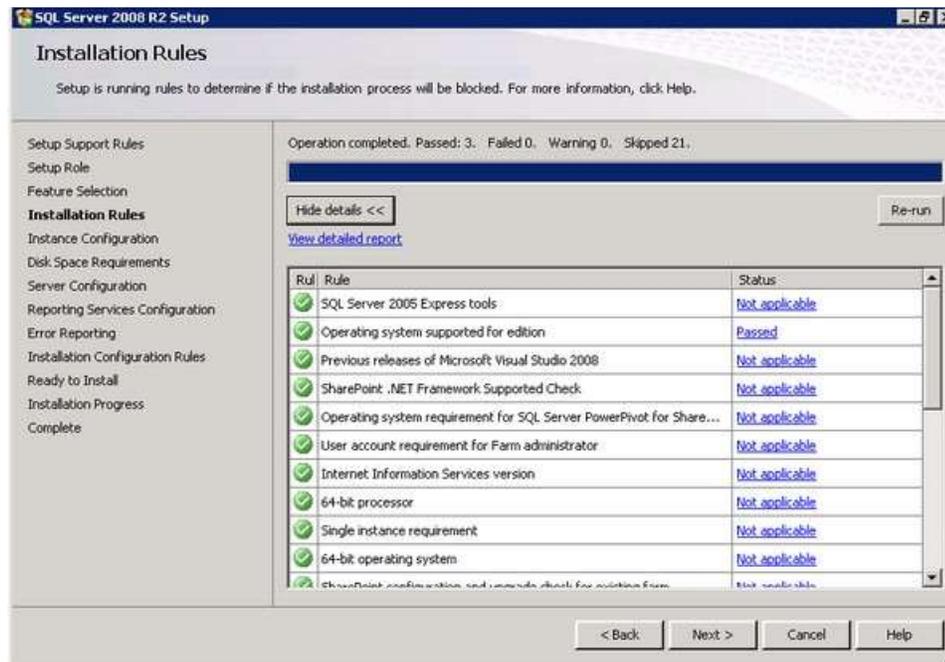
6. In the **Setup Role** dialog box, select the **SQL Server Feature Installation** to install SQL Server 2008 R2 Reporting Services and click **Next**.



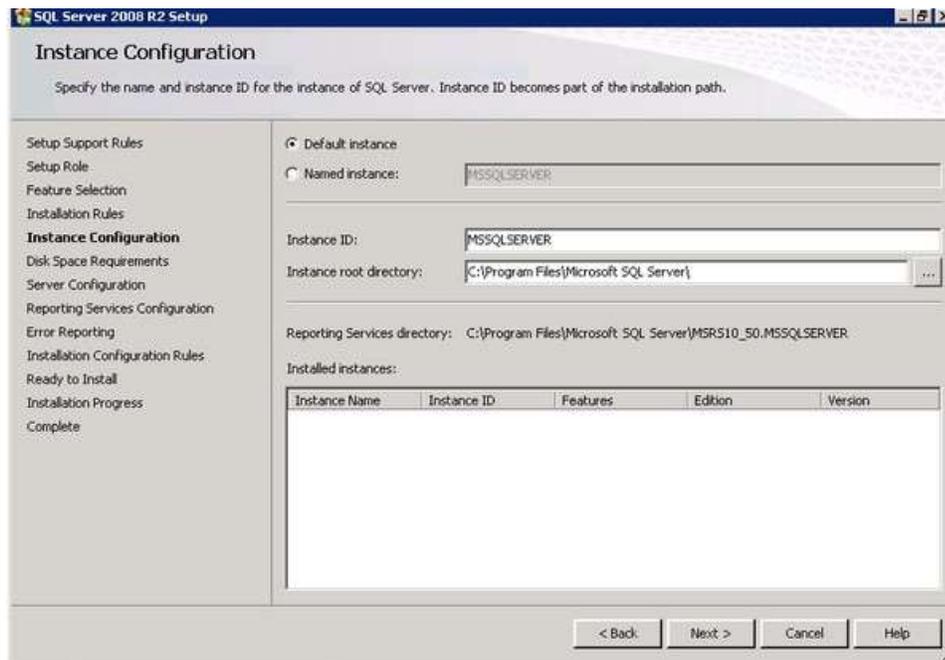
7. In the **Feature Selection** dialog box, select the **Reporting Services** checkbox to install SQL Server 2008 R2 Reporting Services and click **Next**. You can install the client tools and other components as well but, as a best practice, only install those that you need to minimize surface area as well as the components that you need to patch later on.



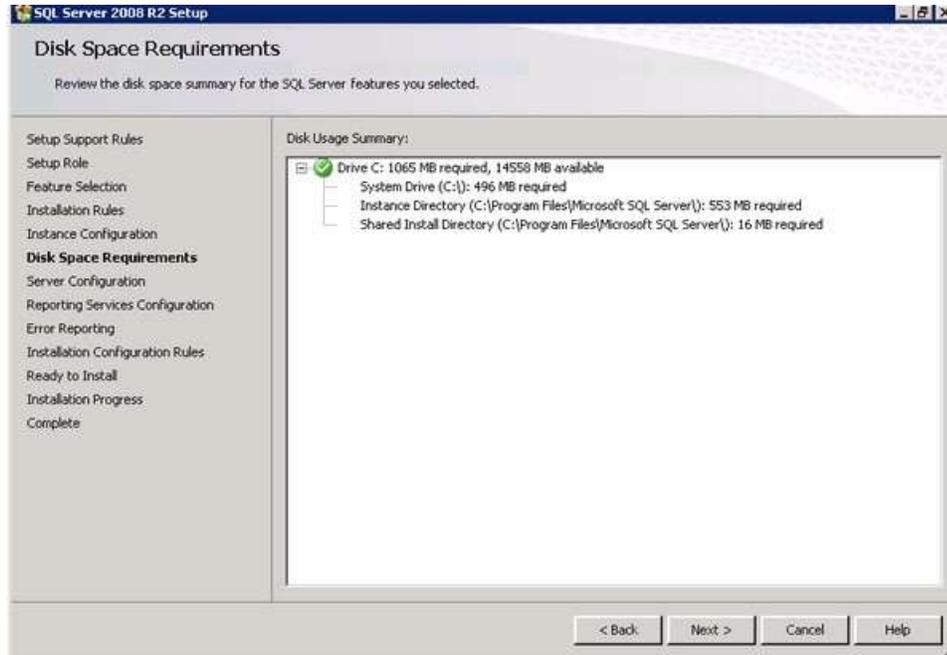
8. In the **Installation Rules** dialog box, validate that the next round of checks return successful results and click **Next**.



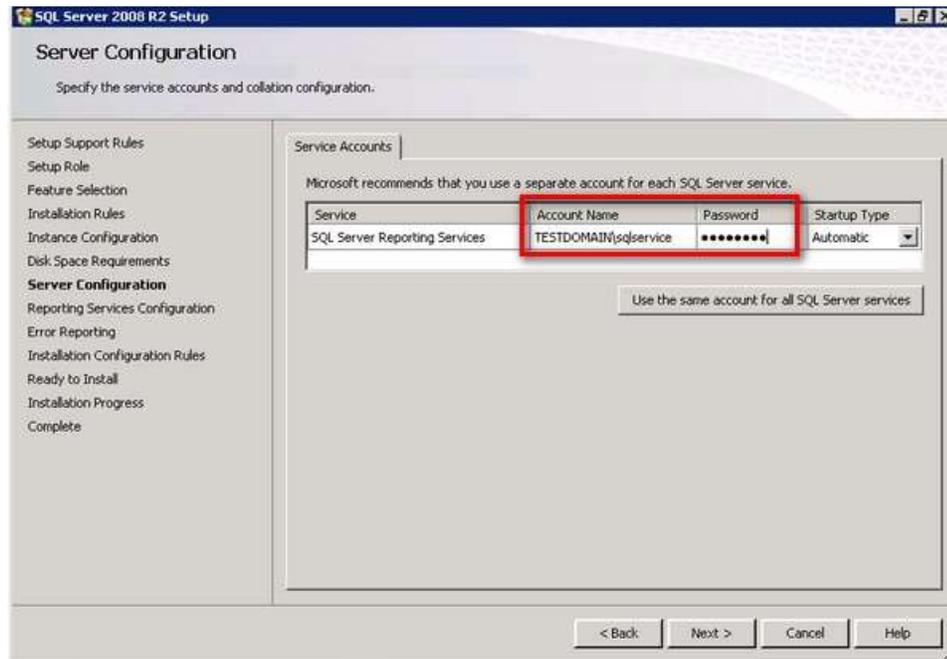
- In the **Instance Configuration** dialog box, select either the **Default instance** or **Named instance**, depending on your custom configuration and click **Next**. For this example, use the default instance



10. In the **Disk Requirements** dialog box, click **Next**.

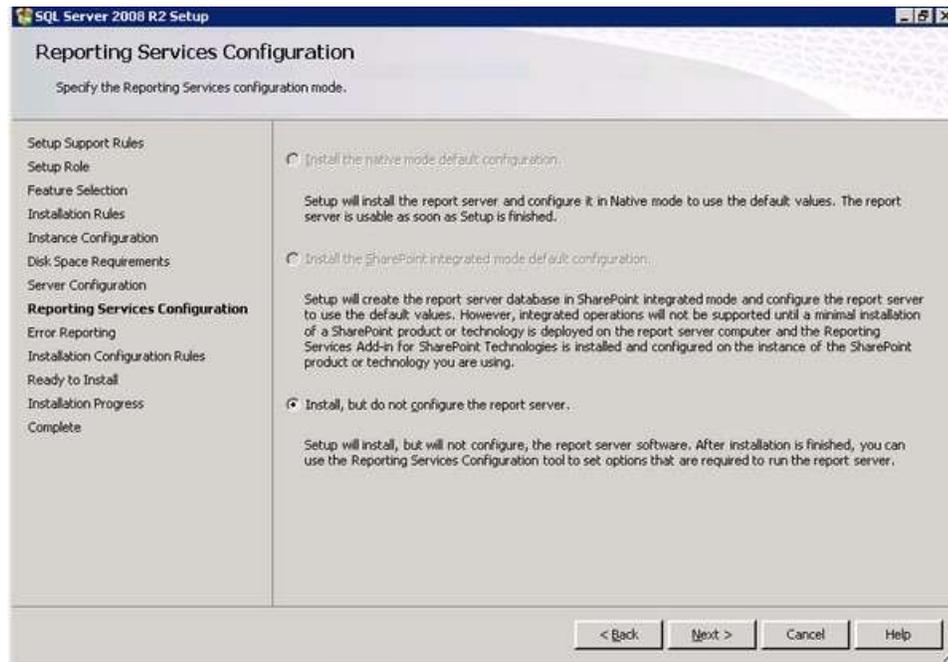


11. In the **Server Configuration** dialog box, enter the service account that you will use for the Reporting Services instance. Click **Next**.

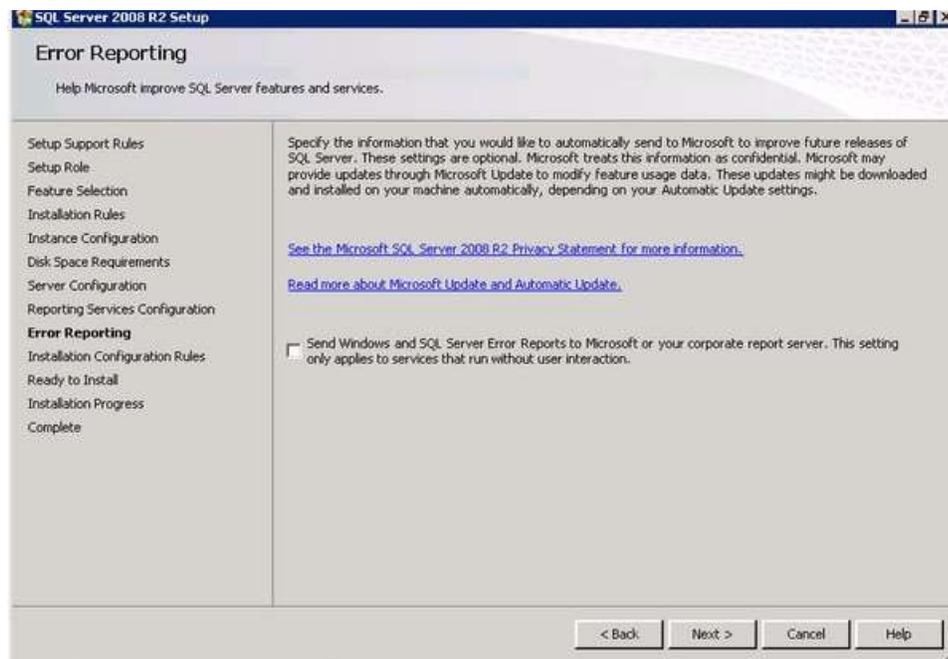


12. In the **Reporting Services Configuration** dialog box, select the **Install, but do not configure the report server** option. This is what Microsoft calls the "*files-only*" mode installation for SQL

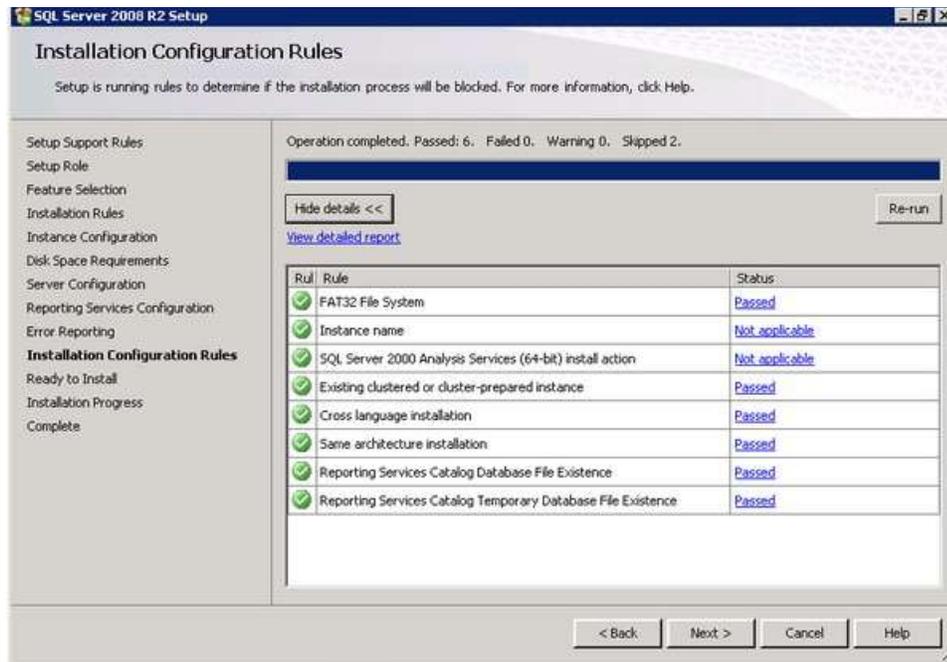
Server 2008 R2 Reporting Services. You will configure the report server after installation using the Reporting Services Configuration Manager. Click **Next**.



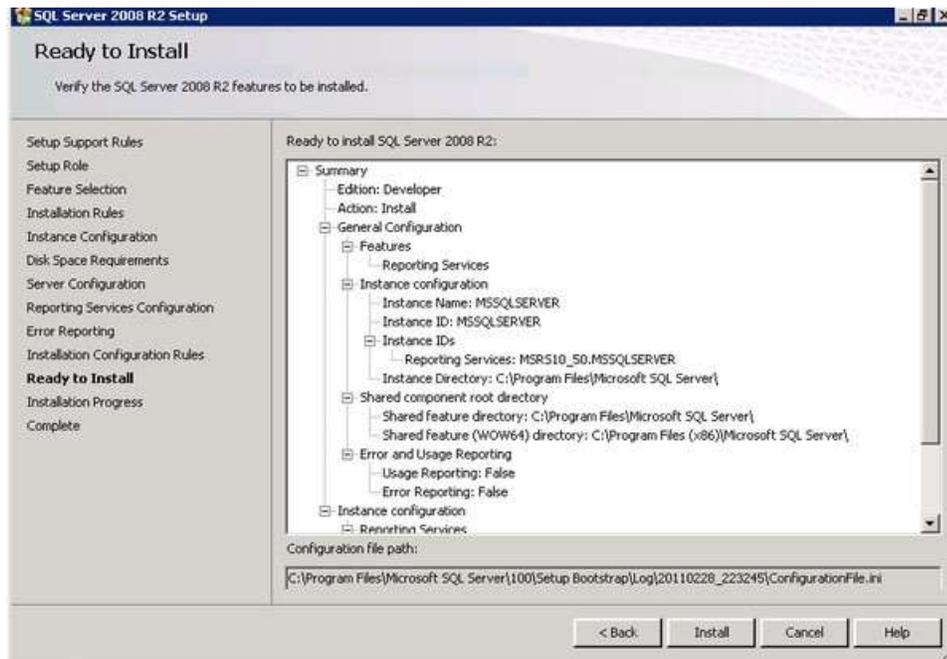
13. In the **Error Reporting** dialog box, click **Next**.



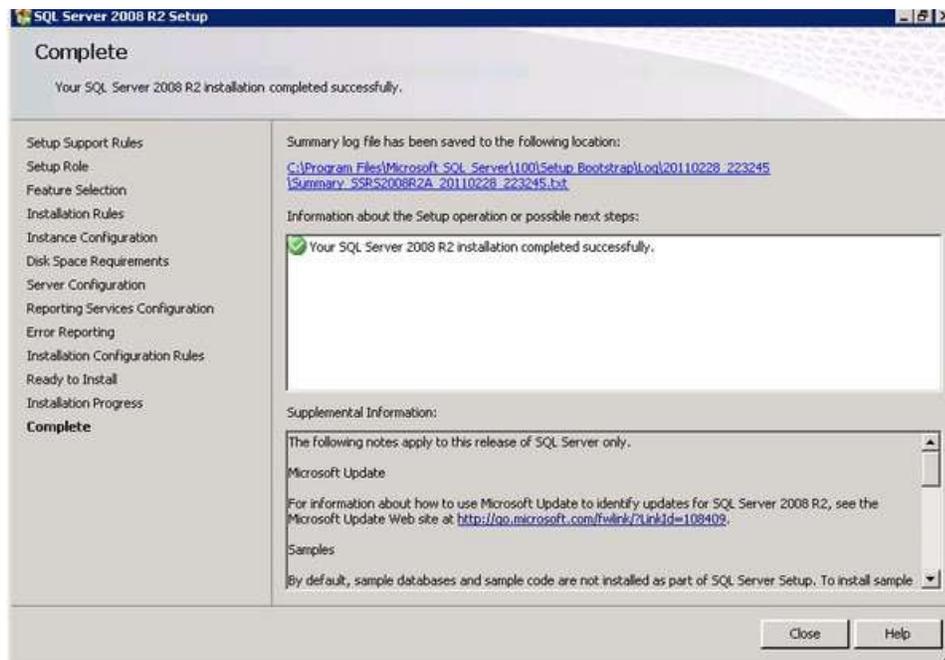
14. In the **Installation Configuration Rules** dialog box, review all the validation checks, making sure that they all return successful results. Click **Next**.



15. In the **Ready to Install** dialog box, review all the configuration settings and click **Install** to proceed with the installation.



16. In the **Complete** dialog box, click **Close** to complete the installation.



Before proceeding to the next part of the installation, make sure that you have successfully installed the SQL Server 2008 R2 Reporting Services binaries on all of the nodes of the NLB cluster and that the SQL Server instance that will host the report server database has been configured for remote connections.

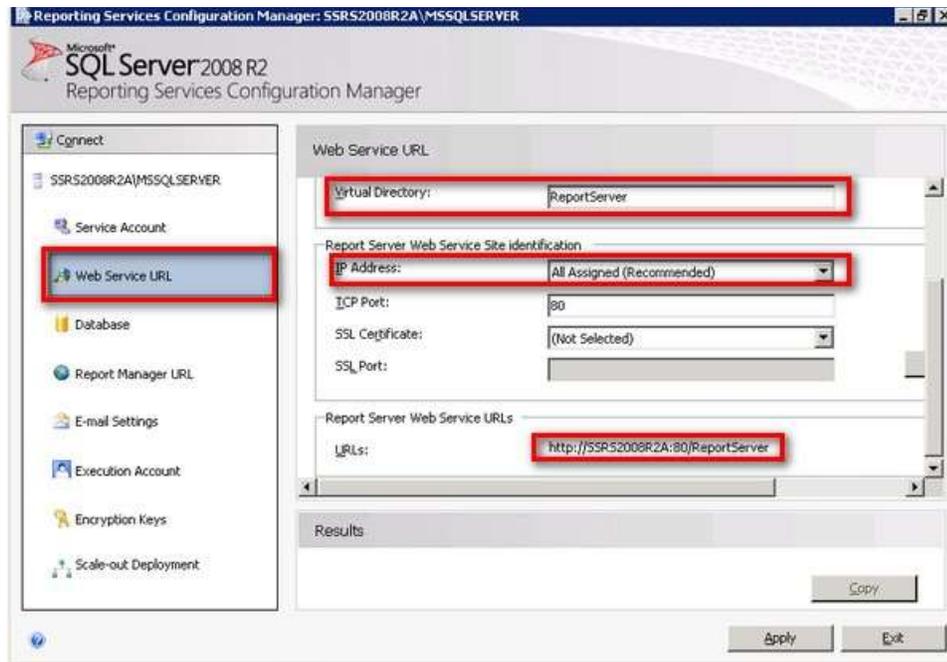
## Configuring the first SQL Server 2008 R2 Reporting Services instance

You will use the Reporting Services Configuration tool to connect the first report server to the shared database. You will have to do this for all of the reporting services instances that you have installed as part of the NLB cluster. For this example, begin with the first SQL Server 2008 R2 Reporting Services instance running on the server **SSRS2008R2A**

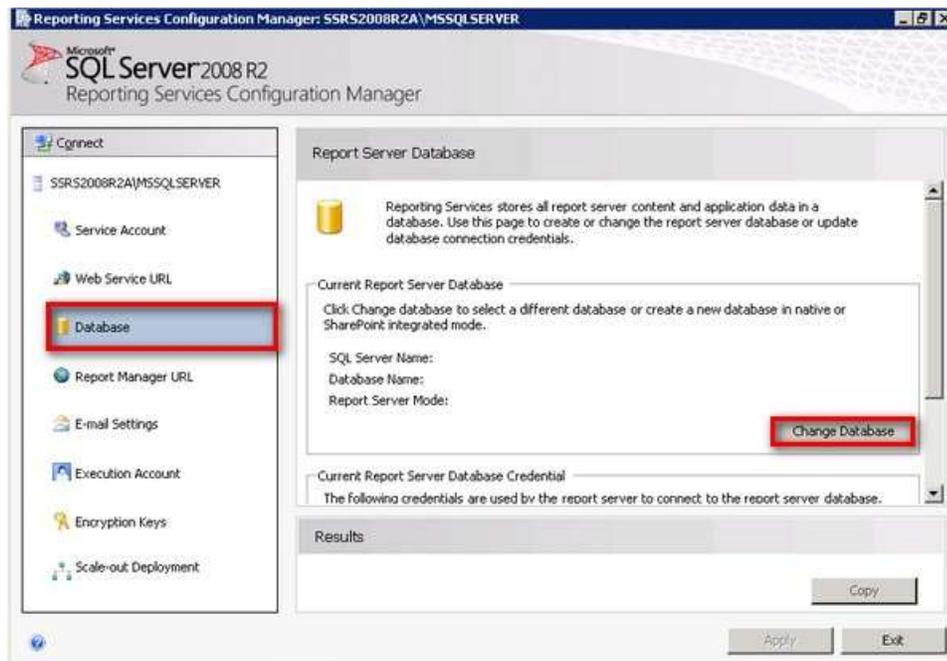
1. On the **Reporting Services Configuration Connection** dialog box, enter the Server Name and Report Server Instance name and click **Connect**.



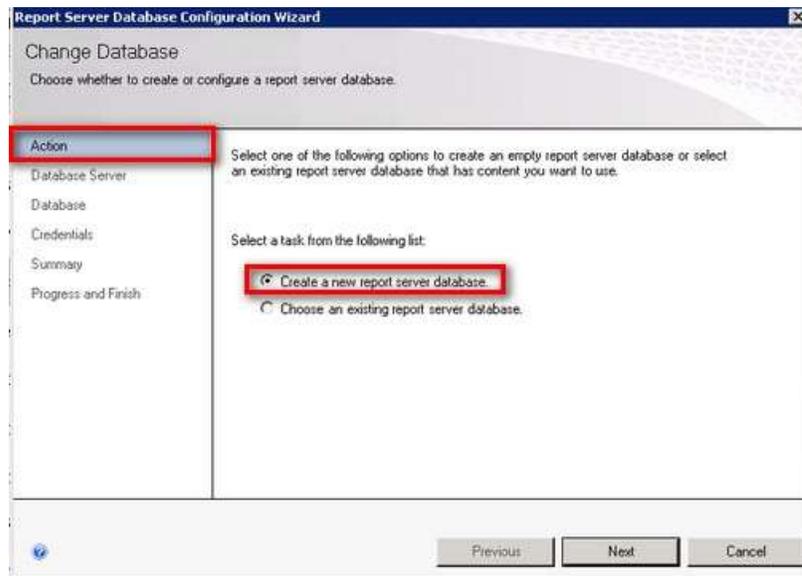
2. Click the **Web Service URL** link. Verify the following values and click **Apply**.
  - **Virtual Directory:** *ReportServer* - The virtual directory name identifies which web application receives the request. Because an IP address and port can be shared by multiple applications, the virtual directory name specifies which application receives the request. This is the default value
  - **IP Address:** *All Assigned (Recommended)* - This option specifies that any of the IP addresses that are assigned to the computer can be used in a URL that points to a report server application. This value also considers friendly host names (such as computer names) that can be resolved by a DNS server to an IP address that is assigned to the server. Since this reporting services instance will be a part of the NLB cluster, we will use this value and allow the NLB cluster to manage the URL-to-IP address resolution for the incoming traffic
  - **URLs:** *http://SSRS2008R2A:80/ReportServer* - This will be the URL to access the reporting services instance on this server. As this server is a part of the NLB cluster, the NLB will be responsible to map the URL *http://SSRS2008R2NLB:80/ReportServer* to this URL



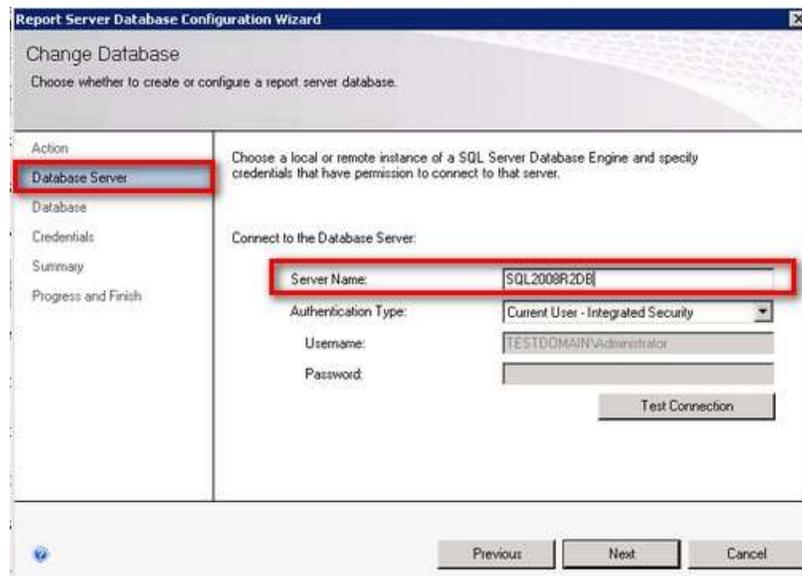
3. Click the **Database** link and click the **Change Database** button. Doing so opens the **Report Server Database Configuration Wizard**



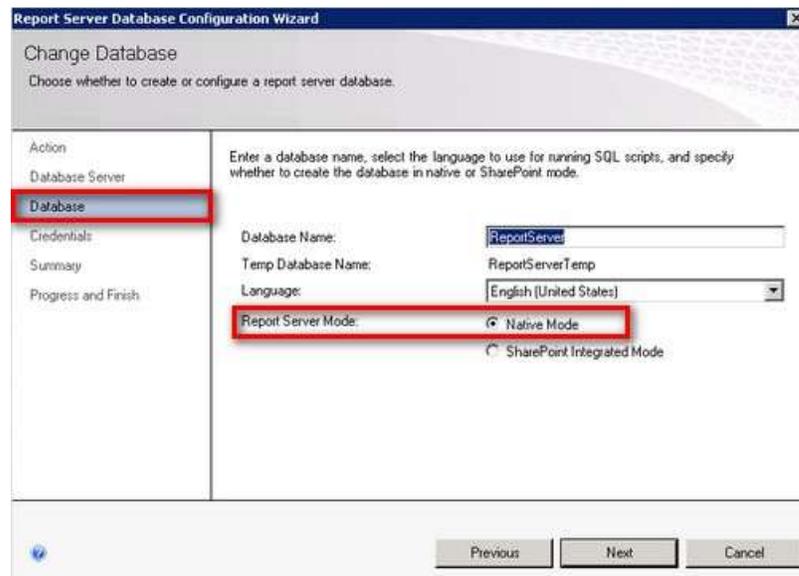
On the **Actions** page, select the **Create a new report server database** option and click **Next**



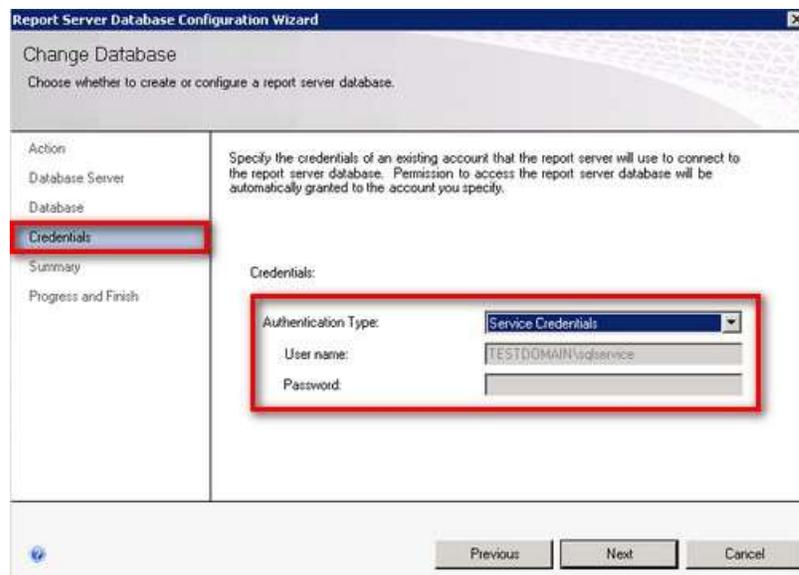
On the **Database Server** page, enter the SQL Server database server instance name in the **Server Name** field. In this example, the SQL Server database instance name is **SQL2008R2DB**. Click **Next**,



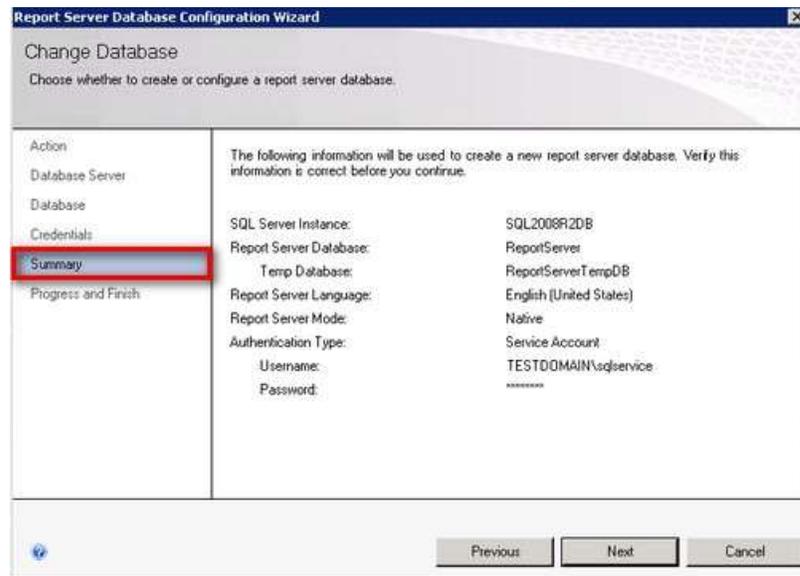
On the **Database** page, accept the default database name **ReportServer** and, under the **Report Server Mode** option, select **Native Mode** (you can also choose SharePoint Integrated if you want this report server instance to work with SharePoint) under the **Report Server Mode** option. Click **Next**.



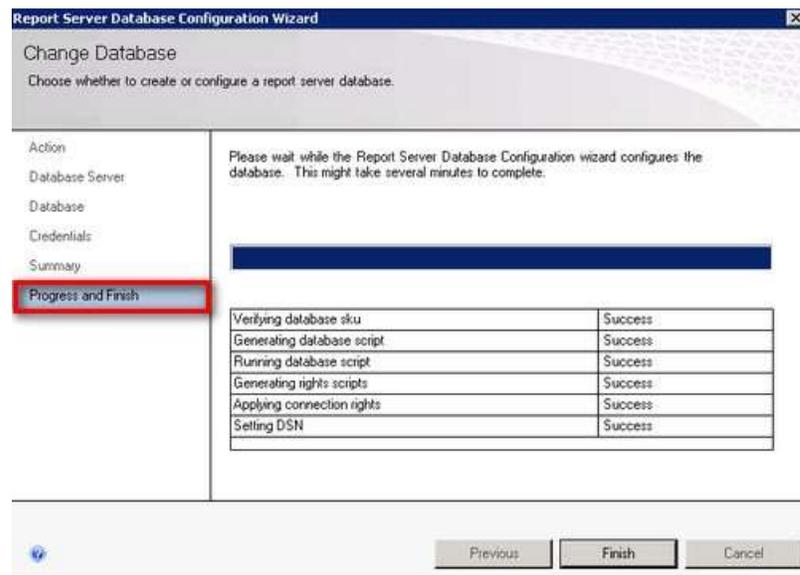
On the **Credentials** page, select the authentication type that you want to use to allow the report server instance to connect to the report server database. In this example, use the service account that you used for the reporting services instance. Click **Next**.



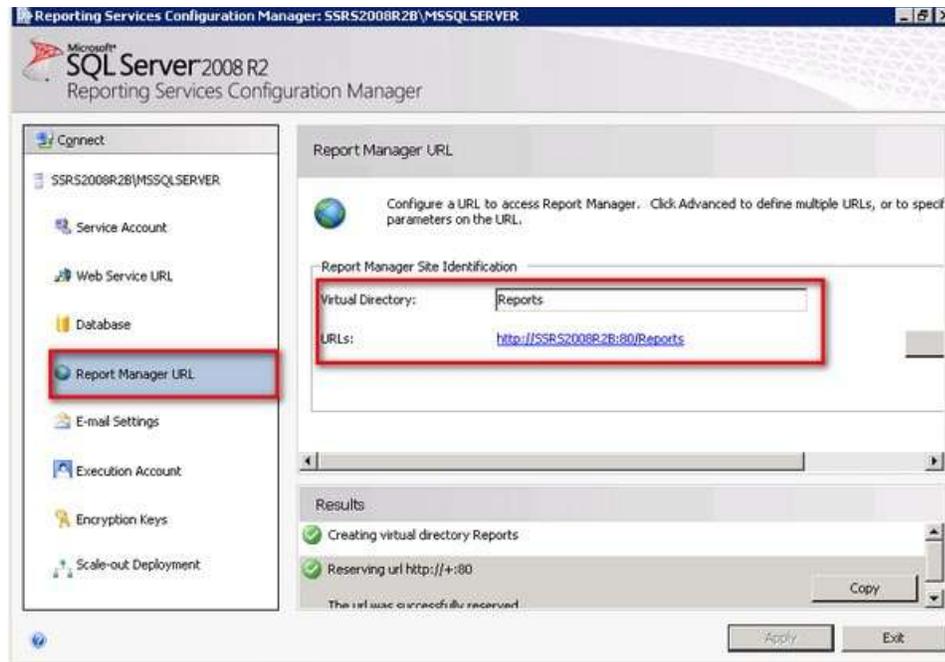
In the **Summary** page, review all of the Reporting Server Database configuration settings. Click **Next** to create the report server database using the credentials that you have provided earlier.



On the **Progress and Finish** page, verify that all the processes completed successfully. If you encounter issues in any of the steps, make sure you resolve them prior to continuing. The two most common issues in creating the report server database are credential permissions and remote connectivity. Click **Finish** to go back to the **Database** page of SQL Server Reporting Services Configuration Manager.



4. Click the **Report Manager URL** link. Accept all the default values and click **Apply**.



5. Verify that the reporting services instance is configured and working properly. You can do this by opening a web browser and accessing the URL defined in the **Web Service URL** page

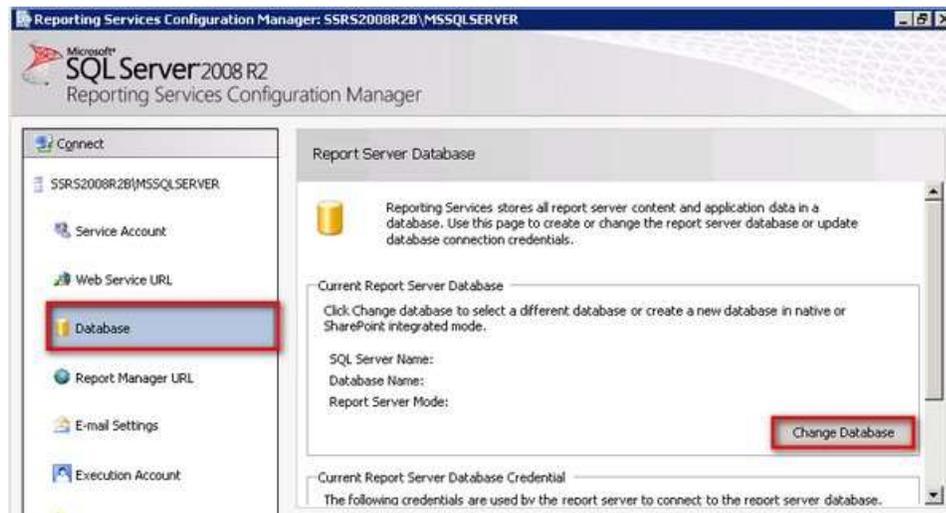


Once you have a working SQL Server 2008 R2 Reporting Services installation working for the first node, we will repeat steps 1 - 4 to add the second node to the NLB cluster.

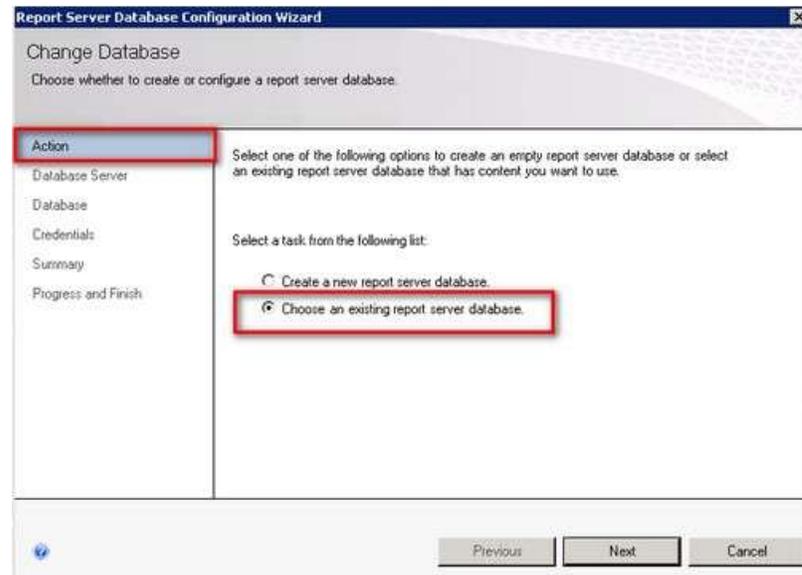
## Configuring the second SQL Server 2008 R2 Reporting Services instance

Before you proceed with configuring the SQL Server 2008 R2 Reporting Services instance on the second node, make sure that you already have it installed in files-only mode as defined in the first server. You'll use the Reporting Services Configuration tool to connect the second report server instance named SSRS2008R2B to the shared database. Follow steps 1 to 4 in the earlier section "*Configuring the first SQL Server 2008 R2 Reporting Services instance*," and accept the default values for the **Web Service URL** and **Report Manager URL** pages. The only difference here is step 3 because you already have an existing report server database. All you have to do is connect the second instance to it.

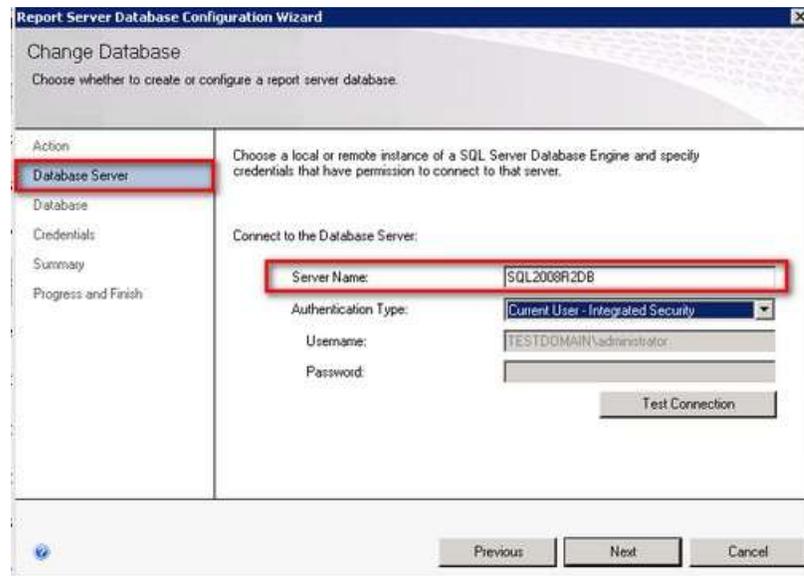
1. Click the **Database** link and click the **Change Database** button to launch the **Report Server Database Configuration Wizard**



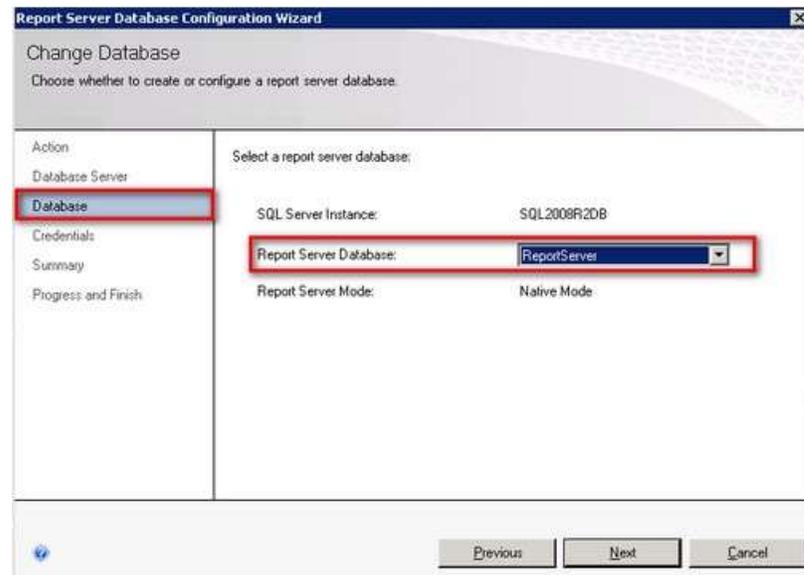
2. On the **Actions** page, select the **Choose an existing report server database** option and click **Next**



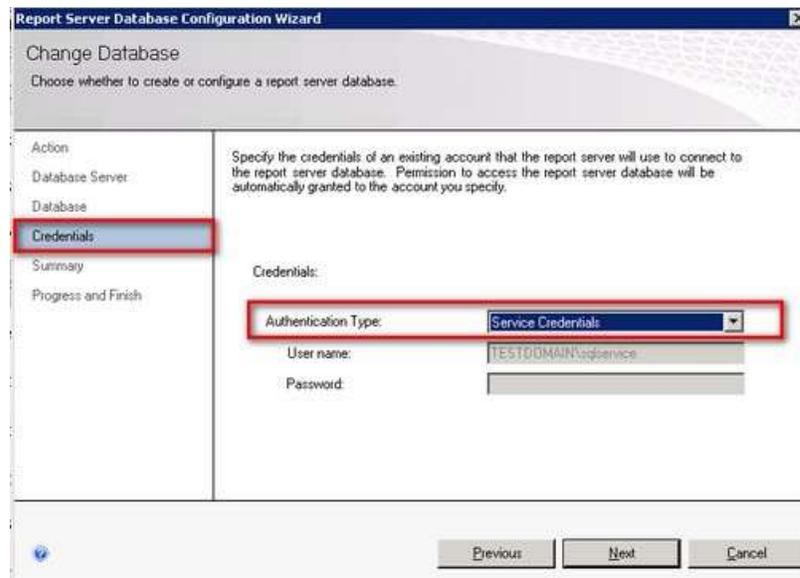
3. On the **Database Server** page, enter the SQL Server database server instance name in the **Server Name** field. This should be the same as what you used in the first reporting services instance. For this example, the SQL Server database instance name is **SQL2008R2DB**. Click **Next**,



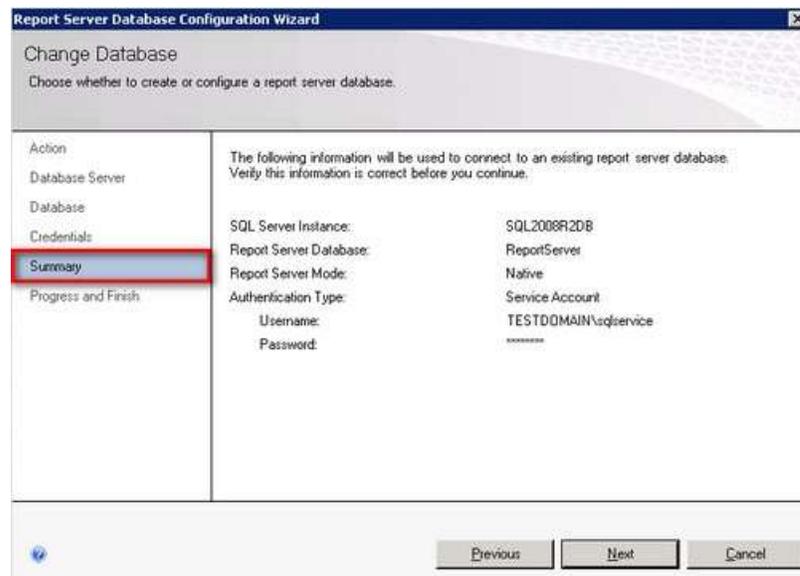
4. On the **Database** page, select the name of the database you used to configure the first report server instance. From the first Reporting Services instance, choose the database name **ReportServer** from the drop-down list. The **Report Server Mode** option value will automatically be filled in based on the report server database selected. Click **Next**.



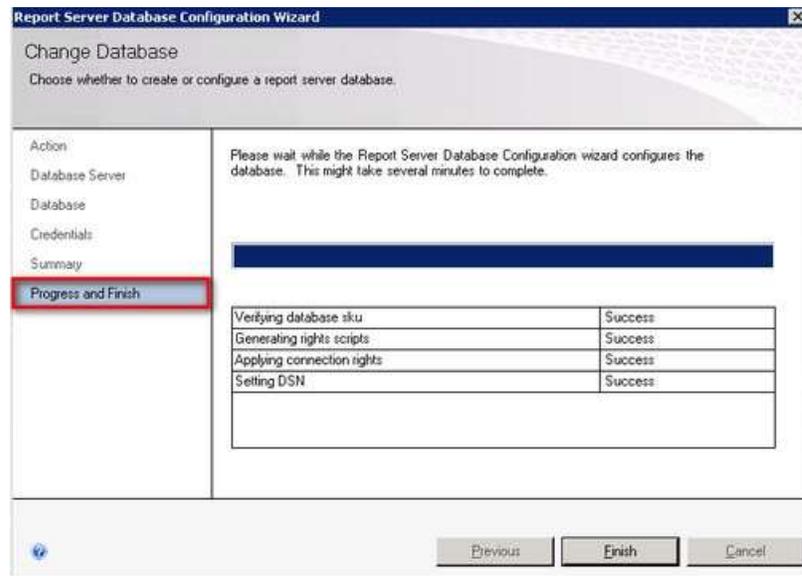
5. On the **Credentials** page, use the same values from the first Reporting Services instance. Choose **Service Credentials** as the Authentication Type as per the previous configuration. Click **Next**.



6. On the **Summary** page, review all of the Reporting Server Database configuration settings. Note that this should be the same as the first Reporting Services instance that you configured. Click **Next**.



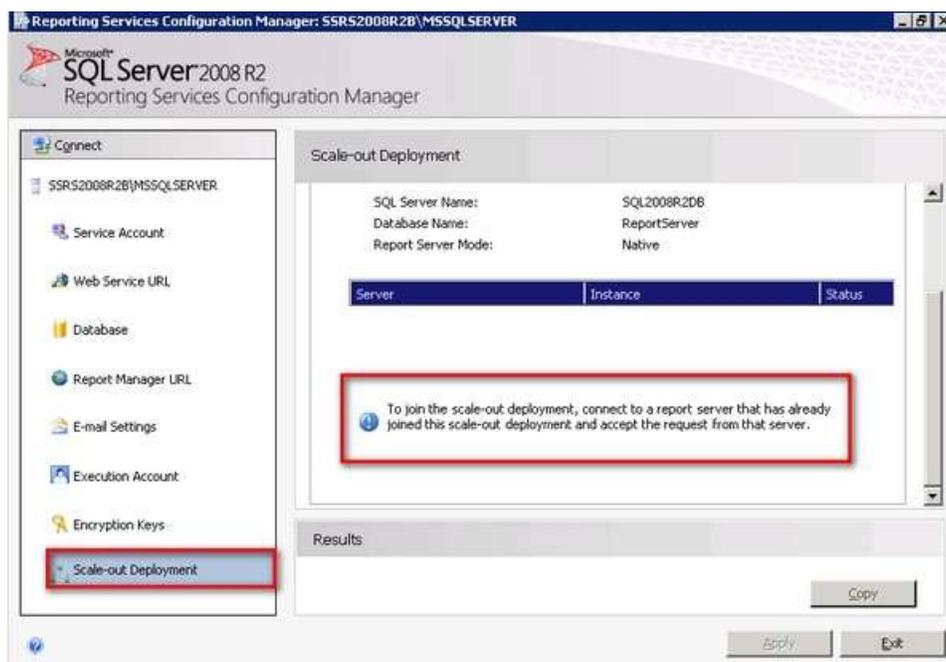
7. On the **Progress and Finish** page, verify that all the processes completed successfully. Click **Finish** to go back to the **Database** page of SQL Server Reporting Services Configuration Manager.



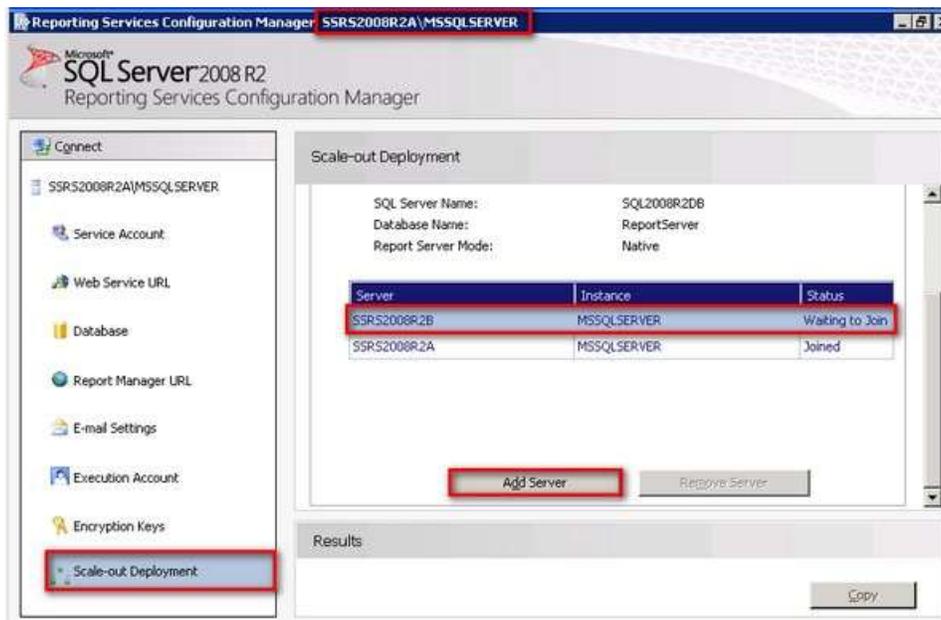
Although step 5 from “*Configuring the first SQL Server 2008 R2 Reporting Services instance*” talks about verifying that the reporting services instance is working properly, you won’t do it this time around. The report server will be unavailable until the report server instance is joined to the scale-out deployment.

## Joining the second SQL Server 2008 R2 Reporting Services instance to the Scale-Out Deployment

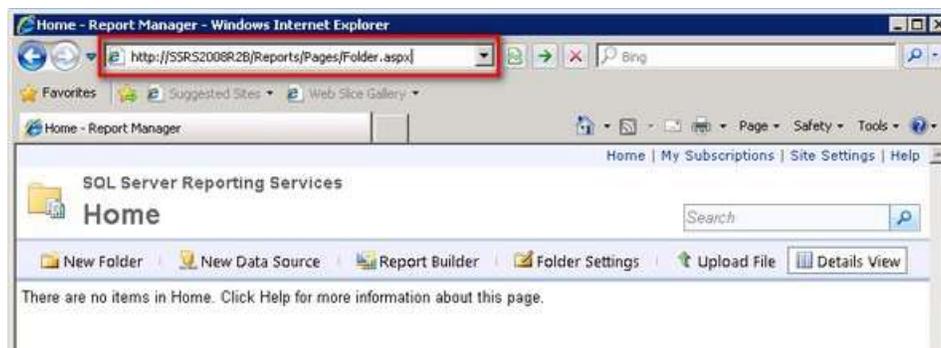
Up to this point, you may not have paid attention to the last link in the Reporting Services Configuration Manager - the **Scale-Out Deployment** link. When you click on that link, you should see a message stating that you should connect to the report server that has already been joined to this scale-out deployment.



This means that we need to connect to the first Reporting Services instance that we configured and accept the second Reporting Services instance to join the scale-out deployment. For this example, you run the Reporting Services Configuration Manager and connect to **SSRS2008R2A**.



You should see the second Reporting Services instance with a **Waiting to Join** status. Select the report server instance that is waiting to join the scale-out deployment and click on the **Add Server** button. You can now verify that both Reporting Services instances are working properly by using the Web Service URL field values to test the Reporting Services individually as if they were still standalone instances.



## Configuring view state validation

The web service URLs work fine after testing access, but you still need to take care of a few things. Note that SQL Server 2008 R2 Reporting Services still uses ASP.NET in its core platform but without IIS. You also need to remember that, because Reporting Services is a web application, the web is stateless. This means that once a web page is processed on the server and rendered on the browser, the server no longer remembers the page anymore. In order for end users to be able to view web pages (or, in this case, HTML reports) interactively, some form of state management must occur.

This can be done by ASP.NET's view state validation. A full discussion of ASP.NET View State is beyond the scope of this document, but I'll discuss a few things to move forward.

By default, view state validation in ASP.NET is enabled and uses automatically generated keys for both validation and encryption. For Reporting Services, view state validation uses the identity of the Report Server web service to perform the validation. But because you're dealing with multiple machines in a NLB cluster, there will be multiple Reporting Services instances and therefore multiple Report Server web service identities to deal with. This would mean that you can't simply rely on a single process identity to perform the view state validation.

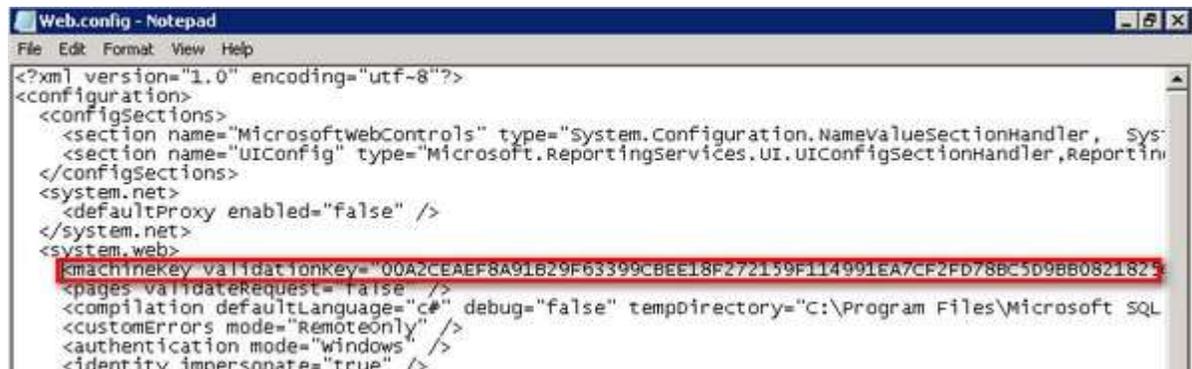
To work around this situation, you can manually generate both the validation and encryption keys instead of relying on the auto generated values provided by ASP.NET. You'll use these key values across all of the nodes in the NLB cluster. You can do so by modifying the *web.config* files for both the report server as well as for the Report Manager. [Microsoft KB article 312906](#) describes how you can create a .NET application that will generate these key values, but I'm not about to open up Visual Studio just to write my own utility. Luckily, there are a lot of utilities out there that will do this for you. I've used a *machineKey* generator utility from <http://aspnetresources.com> to generate the key values for me.

To update the validation and encryption keys, follow these steps:

1. Using any text editor, open the **web.config** file for the Report Manager. By default, this can be found in **\Program Files\Microsoft SQL Server\MSRS10\_50.MSSQLSERVER\Reporting Services\ReportManagerWeb.config**
2. Look for the **<system.web>** section and paste the **<machineKey>** element that you generated either from the machineKey generator utility from aspnetresources.com or your very own utility. The values I generated from the machineKey generator utility from aspnetresources.com is shown below and I pasted it immediately after the **<system.web>** section just so I can easily find it later in case I need to do some troubleshooting.

**<machineKey**

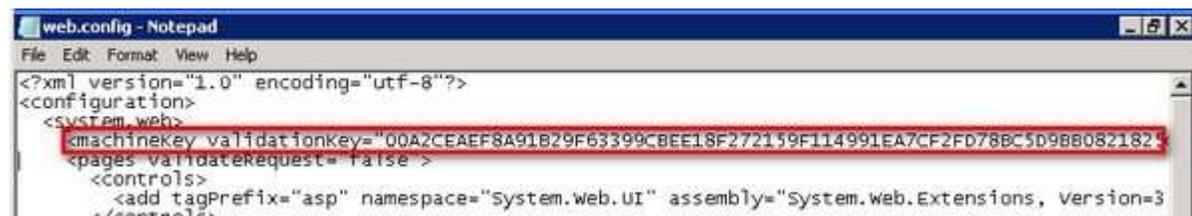
```
validationKey="00A2CEAEF8A91B29F63399CBEE18F272159F114991EA7CF2FD78BC5D9BB0821825C7332C4A4C1698FA58E39634365A97DA8F720377B84F471A3A166CFD31DF"  
decryptionKey="009CA6A1D48DC4DB59E54865C470DFB75FBC1B73AA4833523C9795B1FA88CBE3"  
validation="SHA1" decryption="AES" />
```



The screenshot shows a Notepad window titled "Web.config - Notepad". The XML content is as follows:

```
<?xml version="1.0" encoding="utf-8"?>  
<configuration>  
  <configSections>  
    <section name="MicrosoftWebControls" type="system.Configuration.NameValueCollection, System.Web, Version=2.0.50727.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" />  
    <section name="UIConfig" type="Microsoft.ReportingServices.UI.UIConfigSectionHandler, ReportingServices, Version=10.0.0.0, Culture=neutral, PublicKeyToken=89646d72cf748a74" />  
  </configSections>  
  <system.net>  
    <defaultProxy enabled="false" />  
  </system.net>  
  <system.web>  
    <machineKey validationKey="00A2CEAEF8A91B29F63399CBEE18F272159F114991EA7CF2FD78BC5D9BB0821825C7332C4A4C1698FA58E39634365A97DA8F720377B84F471A3A166CFD31DF" decryptionKey="009CA6A1D48DC4DB59E54865C470DFB75FBC1B73AA4833523C9795B1FA88CBE3" validation="SHA1" decryption="AES" />  
    <pages validateRequest="false" />  
    <compilation defaultLanguage="c#" debug="false" tempDirectory="c:\Program Files\Microsoft SQL Server\MSRS10_50.MSSQLSERVER\Reporting Services\ReportManager\Temp" />  
    <customErrors mode="RemoteOnly" />  
    <authentication mode="Windows" />  
    <identity impersonate="true" />  
  </system.web>  
</configuration>
```

3. Save the **Web.config** file
4. Repeat the previous steps for the Report Server's **web.config** file. By default, this can be found in **\Program Files\Microsoft SQL Server\MSRS10\_50.MSSQLSERVER\Reporting Services\ReportServer\web.config**



The screenshot shows a Notepad window titled "web.config - Notepad". The XML content is as follows:

```
<?xml version="1.0" encoding="utf-8"?>  
<configuration>  
  <system.web>  
    <machineKey validationKey="00A2CEAEF8A91B29F63399CBEE18F272159F114991EA7CF2FD78BC5D9BB0821825C7332C4A4C1698FA58E39634365A97DA8F720377B84F471A3A166CFD31DF" decryptionKey="009CA6A1D48DC4DB59E54865C470DFB75FBC1B73AA4833523C9795B1FA88CBE3" validation="SHA1" decryption="AES" />  
    <pages validateRequest="false" />  
    <controls>  
      <add tagPrefix="asp" namespace="System.Web.UI" assembly="System.Web.Extensions, Version=3.5.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" />  
    </controls>  
  </system.web>  
</configuration>
```

5. Repeat steps 1 to 4 on all of the nodes in the NLB cluster
6. Verify that the **<machineKey>** element in the **<system.web>** section of the **web.config** files for both the Report Manager and Report Server are identical across all of the nodes in the NLB cluster.

## Configuring the Hostname and UriRoot

These configuration items can be found in the *RSReportServer.config* file. The **UriRoot** property is used by the report server delivery extensions to compose URLs that are used by reports delivered in email and/or file share subscriptions. This property has to be set correctly to avoid having incorrect links generated by the reports. Because you're dealing with multiple nodes in the NLB cluster, you wouldn't want end users to see the physical computer name where the subscriptions were generated. You also wouldn't want end users to be directed to the physical server and bypass the NLB cluster because this would affect their session state information.

You need to configure the **UriRoot** property so that the subscribed reports will display the virtual server name of the NLB cluster instead of the physical server names of the NLB cluster nodes. The **Hostname** property defines how the network traffic to the reporting environment is directed. By default, this property isn't defined and will use the physical server name. Remember, you want the network traffic to be directed to the virtual server name so that the NLB cluster will be the one to handle and distribute the requests among the cluster nodes.

To configure the **Hostname** and **UriRoot** properties, follow these steps:

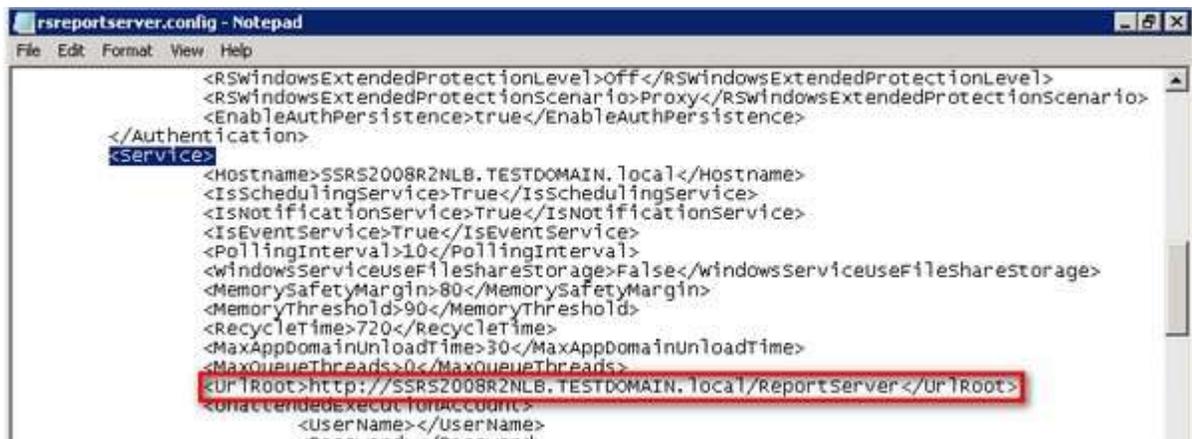
1. Using any text editor, open the **RSReportServer.config** file. By default, this can be found in **\Program Files\Microsoft SQL Server\MSRS10\_50.MSSQLSERVER\Reporting Services\ReportServer\RSReportServer.config**.
2. Look for the **<Service>** section, and add the following information to the configuration file. Use the **Hostname** value of the virtual server name of your NLB server. For this example, use **SSRS2008R2NLB.TESTDOMAIN.local**. Similar to what I've done in the **<machineKey>** element in the *web.config* file, I pasted it immediately after the **<Service>** section just so I can easily find it later on should I need to do some troubleshooting.

```
<Hostname>SSRS2008R2NLB.TESTDOMAIN.local</Hostname>
```



3. Look for the **<UriRoot>** element. By default, this has no value but the default value used is in the format **http://** or **https://<physicalServerName>/<reportserver>**, where **<reportserver>** is the virtual directory name of the Report Server web service.
4. Type a value for **UriRoot** that includes the virtual server name in this format: **http://** or **https://<virtualServerName>/<reportserver>**. For this example, use **http://SSRS2008R2NLB.TESTDOMAIN.local/ReportServer**

```
<UriRoot>http://SSRS2008R2NLB.TESTDOMAIN.local/ReportServer</UriRoot>
```

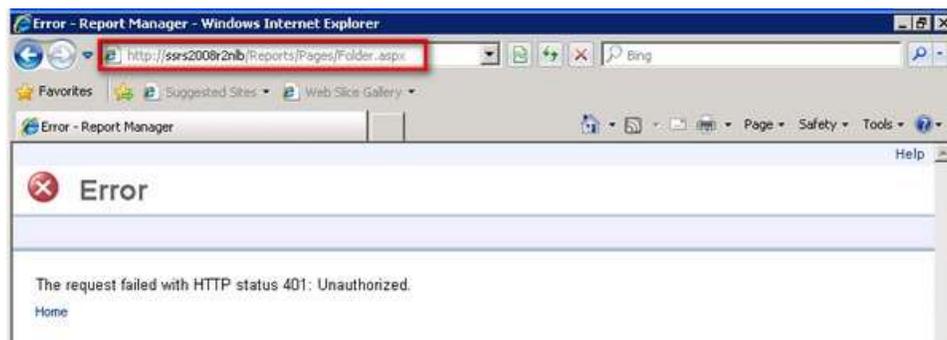


```
<RSWindowsExtendedProtectionLevel>Off</RSWindowsExtendedProtectionLevel>
<RSWindowsExtendedProtectionScenario>Proxy</RSWindowsExtendedProtectionScenario>
<EnableAuthPersistence>true</EnableAuthPersistence>
</Authentication>
<Service>
  <Hostname>SSRS2008R2NLB.TESTDOMAIN.local</Hostname>
  <IsSchedulingService>True</IsSchedulingService>
  <IsNotificationService>True</IsNotificationService>
  <IsEventService>True</IsEventService>
  <PollingInterval>10</PollingInterval>
  <WindowsServiceUseFileShareStorage>False</WindowsServiceUseFileShareStorage>
  <MemorySafetyMargin>80</MemorySafetyMargin>
  <MemoryThreshold>90</MemoryThreshold>
  <RecycleTime>720</RecycleTime>
  <MaxAppDomainUnloadTime>30</MaxAppDomainUnloadTime>
  <MaxQueueThreads>0</MaxQueueThreads>
  <UrlRoot>http://SSRS2008R2NLB.TESTDOMAIN.local/ReportServer</UrlRoot>
  <UnattendedExecutionAccount>
    <UserName></UserName>
    <Password></Password>
  </UnattendedExecutionAccount>
</Service>
```

5. Save the **RSReportServer.config** file
6. Repeat steps 1 to 6 on all of the nodes in the NLB cluster
7. Verify that the **<Hostname>** and **<UrlRoot>** elements in the **<Service>** section of the **RSReportServer.config** files are identical across all of the nodes in the NLB cluster.

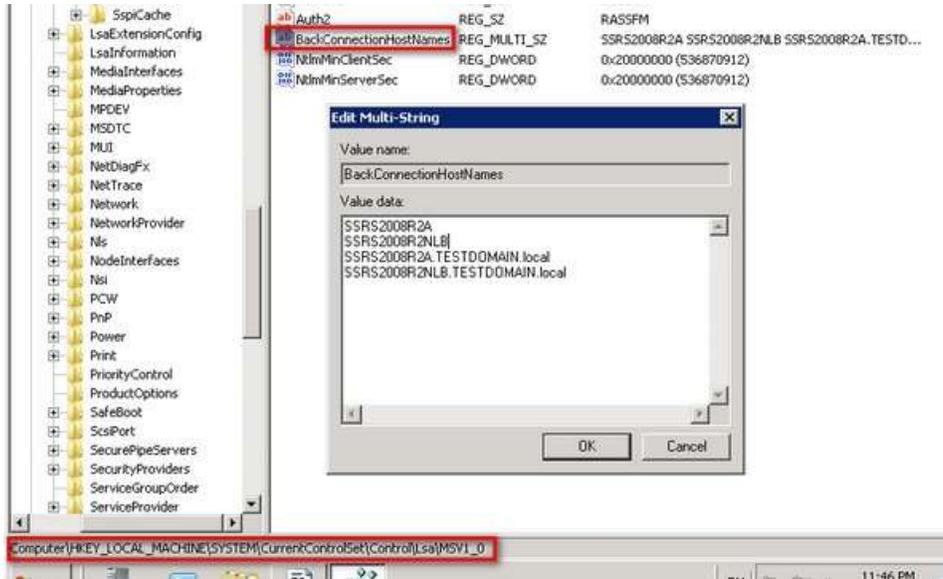
## Workarounds for the HTTP 401 Error Message

At this point, you may be tempted to test access to the Reporting Services instance via the virtual server name. But if you do so, you may get an HTTP 401 error message similar to the one below.

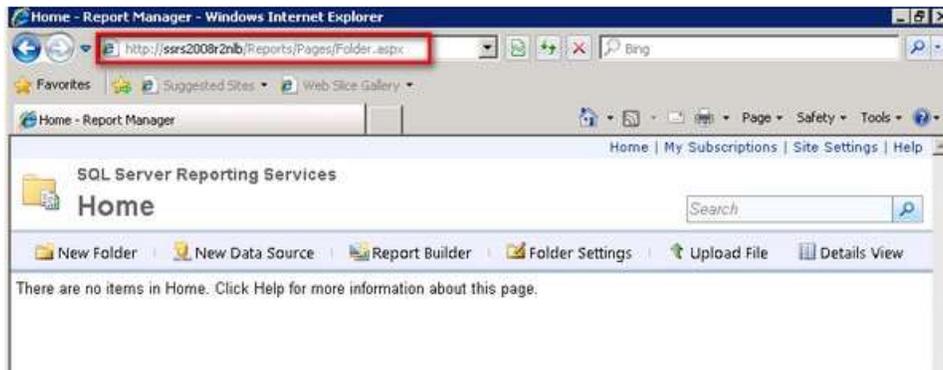


If you look at the web page, you'll notice the user interface associated with SQL Server Reporting Services, which tells you that the Reporting Services instance is working except that it encounters an HTTP 401 error. According to [Microsoft KB article 896861](http://msdn.microsoft.com/en-us/library/ms175548.aspx), this behavior is by design if the fully qualified domain name or the custom host header doesn't match the local computer name, which is the case in a Reporting Services scale-out implementation. And although the Microsoft KB article describes the behavior for a web application running on IIS, it does apply to our scenario. You can implement any of the methods described in the KB article, but Method 1 is the approach Microsoft recommends. You'd need to reboot your server in order for the changes to take effect immediately because you don't have IIS on the NLB cluster nodes. Your **BackConnectionHostNames** key would look something like this for one of the nodes, considering including both the hostnames and the fully qualified domain names for the physical computer name as well as the virtual server name:

SSRS2008R2A  
SSRS2008R2NLB  
SSRS2008R2A.TESTDOMAIN.local  
SSRS2008R2NLB.TESTDOMAIN.local



Make sure you make this registry change across all of the nodes in your NLB cluster, making the necessary changes to the **Value Data** box to reflect the corresponding physical server name for each node. Once all of the nodes in the NLB cluster have been rebooted, you should now be able to access the reporting services instance using the virtual server name.



Congratulations! You now have a highly available, scale-out SQL Server 2008 R2 Reporting Services farm running on a two-node Windows network load-balanced cluster. Now you can add your reports and data sources.

## About The Author



**Edwin Sarmiento** is a Microsoft SQL Server MVP from Ottawa, Canada specializing in high availability, disaster recovery and system infrastructures running on the Microsoft server technology stack. He is very passionate about technology but has interests in music, professional and organizational development, leadership and management matters when not working with databases. He lives up to his primary mission statement – *“To help people and organizations grow and develop their full potential as God has planned for them.”*

He wants the whole world to know that the FILIPINO is a world-class citizen and brings JESUS CHRIST to the world.

He blogs at <http://bassplayerdoc.wordpress.com> but you can also follow him on [Twitter](#) : @bassplayerdoc