



SABC Cluster Servers

About	Description
Revision	4.6.5-1.0
History	
Authors	Jerry Clark

Overview

About this manual

The Swift Newsroom system at SABC relies on 4 central servers:

1. **Repository Server** - stores the Subversion Repository for all the Swift projects in use at SABC.
2. **MySQL Server** - stores external data for use in Swift graphics
3. **MOS Gateway Server** - creates and distributes Swift running orders received from ENPS
4. **DataServer Server** - stores details of the Swift Live logins and Sessions

Because these servers are a broadcast-critical part of the infrastructure each one is implemented as a clustered pair of machines with one machine acting as the “Main” server and the other acting as the “Backup” server. Should the “Main” fail at any time the “Backup” will take over in a seamless fashion.

This manual details the setup, configuration and maintenance of each of the four “Clusters” at SABC.

Overview of the SABC Clusters

Each of the clusters is implemented as a Microsoft Active Directory-detached Cluster - Microsoft documentation can be found here:

[https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn265970\(v=ws.11\)#:~:text=An%20Active%20Directory%2Ddetach ed%20cluster%20uses%20Kerberos%20authentication%20for%20intracluster,scenario %20that%20requires%20Kerberos%20authentication.](https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn265970(v=ws.11)#:~:text=An%20Active%20Directory%2Ddetach ed%20cluster%20uses%20Kerberos%20authentication%20for%20intracluster,scenario %20that%20requires%20Kerberos%20authentication.)

An Active Directory-detached failover cluster has no dependencies in Active Directory Domain Services (AD DS) for network names and only requires that the failover cluster nodes are joined to an Active Directory domain.

Each cluster also requires a Cluster Quorum Witness to be configured. For more details see:

<https://learn.microsoft.com/en-us/azure-stack/hci/concepts/quorum>

Each of the 8 cluster “nodes” has been joined to an SABC AD domain and because these are “AD–detached” clusters the clusters themselves can only be created via Windows Powershell commands.

Each of the cluster nodes is also configured to run a Powershell script every 5 seconds. This script updates a file (C:\ProgramData\RTSoftware\RTSW\conf\isBackupServer.txt). If this file contains a “1” then the node is the “Backup” server. If this file contains a “0” then the node is the “Main” (or Active) server.

SABC have four Clusters, each one comprising a pair of HP zCentral 4R rack workstations running Windows Server 2019.

The following table shows the names/IP addresses for each cluster:

Cluster Name	Host Node DNS Name	Host Node IP Address	Cluster DNS Name	Cluster IP Address
MOS Gateway	NSR-MOS1-RND-S	10.235.71.121	NSR-MOS-RND-C	10.235.71.100
	MCR-MOS2-BAC-S	10.235.71.64		
Data Server	NSR-NRCS1-SVR	10.235.71.122	NSR-NRCS-SVR-C	10.235.71.101
	MCR-NRCS-BAC	10.235.71.65		
Media Watcher	NSR-SM-MW1	10.235.71.123	NSR-SM-MW-C	10.235.71.102
	MCR-SM-NN-MW2	10.235.71.66		
Repository	NSR-MNG-REP1	10.235.71.124	NSR-MNG-REP-C	10.235.71.103
	MCR-MNG-REPBAC	10.235.71.62		

For Administration purposes each node supports Remote Desktop login. Applications using a service hosted on a cluster should connect to the Cluster IP address/DNS name. Connections to the cluster IP address are serviced by the active node within the cluster pair.

Creating and Configuring a Cluster

Each of the 4 clusters is created and configured in an identical fashion. In this Chapter I will be showing the Mos Gateway cluster as an example.

The example Powershell commands should be executed from a Powershell window that has been "Run As" Administrator.

Cluster Creation

The powershell command is (all typed on one line):

```
New-Cluster <cluster-name> -Node <Node-1>,<Node-2>  
-StaticAddress <Cluster IP address> -NoStorage  
-AdministrativeAccessPoint Dns
```

Where:

<cluster name> = the DNS name of the cluster
<Node-1> = the DNS name of the first node in the cluster
<Node-2> = the DNS name of the second node in the cluster
<Cluster-IP> = the IP address of the cluster

Witness Configuration

Each cluster has now been created but requires a "Witness" to be configured. There are several different types of Witness but in this case we will configure a File Share Witness. For further details please see:

<https://learn.microsoft.com/en-us/windows-server/failover-clustering/file-share-witness>

At SABC the File Share is provided by the Facilis system.

The name of the share is \\NSR-FAC-SAN\\W with:

User name: facilis

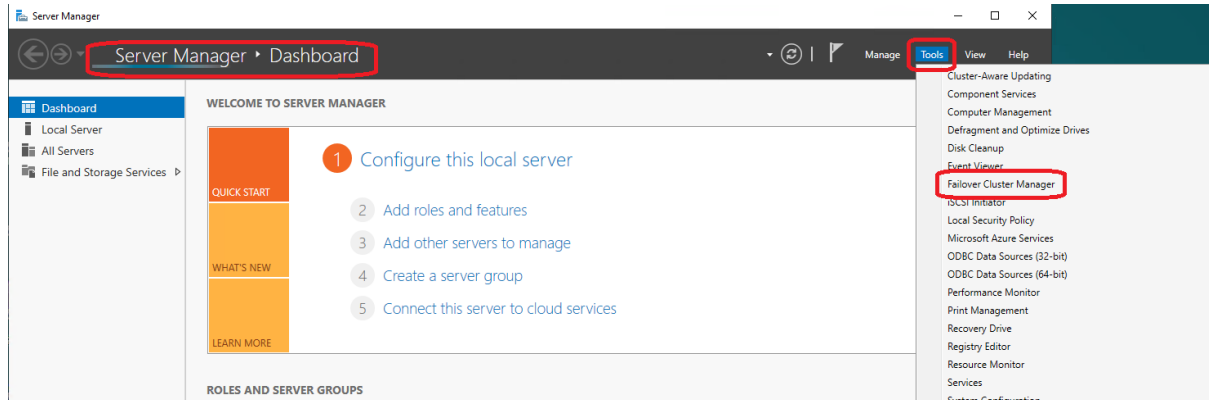
Password: W!ndowsRoot

When the Set-ClusterQuorum command is run it will prompt for the username and password as given above.

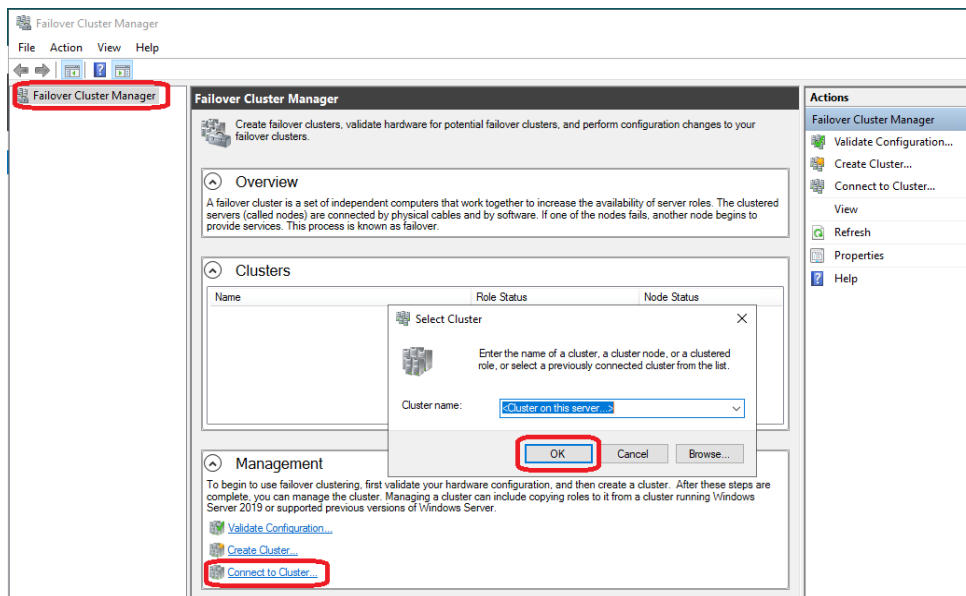
```
Set-ClusterQuorum -FileShareWitness \\NSR-FAC-SAN\\W  
-Credential (Get-Credential)
```

Check Cluster Created Successfully

If the above 2 commands have run successfully we should now be able to run up the “Failover Cluster Manager” from the tools menu on the Server Manager dashboard:



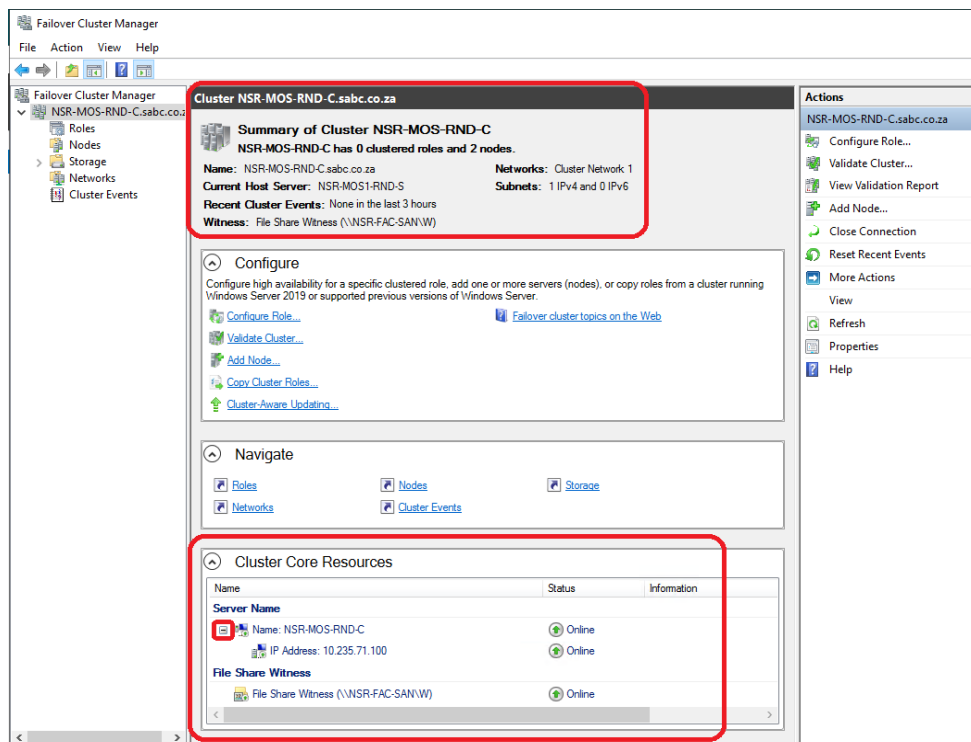
In the Failover CLuster Manager select “Connect to Cluster” then OK:



Once connected to the cluster various bits of information should be shown.

In the top section “Summary of Cluster” you should see the cluster name and the number of nodes. Below that should be the full DNS name of the cluster. The current host is shown (the currently active server in the pair) and there should be no cluster events (errors) shown. Lastly the File Share Witness should be shown.

In the “Cluster Core Resources” section you should see the cluster name and its associated IP address and below that the File Share Witness share name.



Please note that as we are using “AD-detached” clusters the Failover Cluster Manager can be used to query & monitor the cluster but most administration has to be done from Powershell.

Setup Powershell script

For the RTSW Newsroom server software to run correctly it needs to know if it is running on the active server or on the backup server. It does this by querying a set file:

C:\ProgramData\RTSoftware\RTSW\conf\isBackupServer.txt

If this file contained a “0” then this is the currently active server. If this file contains a “1” then this server is currently the backup server.

The file is created and updated by a Powershell script that is called every 5 seconds from a DOS script. The DOS script is run as a Windows Service. Because Windows Services cannot be script files the DOS script is run via NSSM.exe which does allow .bat files to run as a service.

The DOS .bat script is:

To be added

The Powershell script that it calls every 5 seconds is:

To be added

Both of these scripts and the NSSM executable are saved in:

C:\ProgramData\RTSoftware\RTSW\bin

Cluster Deletion

If it ever becomes necessary to delete and reinstate a cluster it has to be done from the Powershell. From a node on the cluster execute Powershell command:

bladwdwdw

which will delete the cluster. After cluster deletion the server should be rebooted.

Individual Cluster Details

The MOS Gateway Cluster

Cluster Name	Host Node DNS Name	Host Node IP Address	Cluster DNS Name	Cluster IP Address
MOS Gateway	NSR-MOS1-RND-S	10.235.71.121	NSR-MOS-RND-C	10.235.71.100
	MCR-MOS2-BAC-S	10.235.71.64		

Overview

Both nodes in this cluster run the RTSW Mos Gateway application that allows ENPS (via an ActiveX plugin) to push Running Orders to this cluster. ENPS pushes the Running Order to both nodes in the cluster and the currently active node will push that running order through to both of the nodes in the Data Server cluster.

MOS Gateway Application Configuration

To be completed

Running Order Distribution

To be completed

The Data Server Cluster

Cluster Name	Host Node DNS Name	Host Node IP Address	Cluster DNS Name	Cluster IP Address
Data Server	NSR-NRCS1-SVR	10.235.71.122	NSR-NRCS-SVR-C	10.235.71.101
	MCR-NRCS-BAC	10.235.71.65		

Overview

This cluster runs the RTSW DataServer application that is accessed by the RTSW Swift Live playout system.

Both nodes run the DataServer application and the active node pushes changes to the backup node to keep the 2 nodes synchronized.

Swift Live is configured to query the Cluster name/IP address and the query will be answered by the currently active node.

DataServer Application Configuration

To be completed

Cluster Node Synchronisation

To be completed

The MediaWatcher Cluster

Cluster Name	Host Node DNS Name	Host Node IP Address	Cluster DNS Name	Cluster IP Address
Media Watcher	NSR-SM-MW1	10.235.71.123	NSR-SM-MW-C	10.235.71.102
	MCR-SM-NN-MW2	10.235.71.66		

Overview

This cluster runs the RTSW MediaWatcher application that is accessed by the RTSW Swift Live playout system. Both nodes also run a MariaDB Database to store the incoming data. On both nodes MariaDB is configured in a master-master configuration which means that changes to the database on one node will be replicated on the other node to keep both databases synchronised.

Both nodes run the MediaWatcher application and process any incoming data. The active node in the cluster will update its local database and these updates will get replicated on the backup node.

Database queries are configured to use the Cluster name/IP address and the query will be answered by the currently active node.

MediaWatcher Application Configuration

To be completed

Cluster Node Synchronisation

To be completed

The Repository Cluster

Cluster Name	Host Node DNS Name	Host Node IP Address	Cluster DNS Name	Cluster IP Address
Repository	NSR-MNG-REP1	10.235.71.124	NSR-MNG-REP-C	10.235.71.103
	MCR-MNG-REPBAC	10.235.71.62		

Overview

This cluster acts as the repository for the RTSW Repository application.

Both nodes run the VisualSVN Subversion server application which is configured in a master-slave configuration. If both nodes are available both the master and slave repositories are available for read and write. If the master repository becomes unavailable then the slave becomes read-only and must be re-configured as a “temporary master” for normal operation to be resumed.

Clients accessing the Subversion repository use the cluster IP address/name and any Subversion query made on the cluster IP address will be served by the currently active node in the cluster.

In normal operation the “master” repository is on the master node in the cluster, the “master” cluster node is the “active” node and any VisualSVN updates will be automatically replicated onto the repository on the backup node. On failover the backup node will become the active node and the repository will become read-only until VisualSVN is reconfigured to become the new “master”.

When the failed node becomes available again VisualSVN should be configured as a slave to the new master at which time it will synchronise with the new master.

VisualSVN Configuration

For VisualSVN installation and configuration please see the Repository manual:

<https://rtsw.co.uk/document/repository-user-guide-installation-and-configuration3-1/>

VisualSVN Failover and Recovery Procedure

As initially setup the Cluster “master” node is the “active” node and is also the VisualSVN “master” repository.

A fault on the cluster master node may cause the cluster to failover. The immediate effect of this will probably be errors when using the Repository application to check-in a new project or changes to an existing project.

If the Repository application cannot update the VisualSVN repository it's very likely that the cluster has failed-over. The failover means the active node will now be the “backup” node in the cluster and it will be running the “slave” VisualSVN repository which will have become read-only when the “master” node on the cluster became unavailable causing the failover. The first step is to verify which cluster node is “active” and whether the VisualSVN Repository is a “master” or a “slave” on the active node.

If the “active” node has a “slave” repository the next step is to use the VisualSVN Server Manager to check if the repository is writable.

If the repository is not writable then a manual intervention is required to make the “slave” VisualSVN repository a “master” and therefore enable reads and writes on the repository. Once this has been done the Repository application should work as normal.

The final step is to restore the failed cluster node (it will now be the passive, non-active node) and create a VisualSVN repository that is a “slave” to the new “master” and on the now active node.

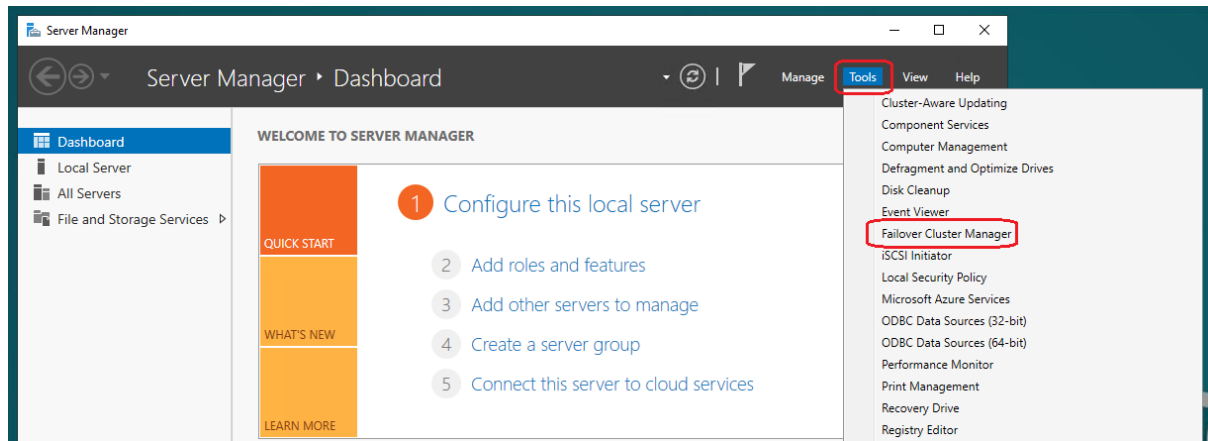
In summary the steps required are:

1. Identify active node and the status of the repository on that node
2. If the repository is a slave check to see if its writable
3. If the repository is not writable make it the new “master”
4. Restore failed node and recreate the repository as a slave to the new master

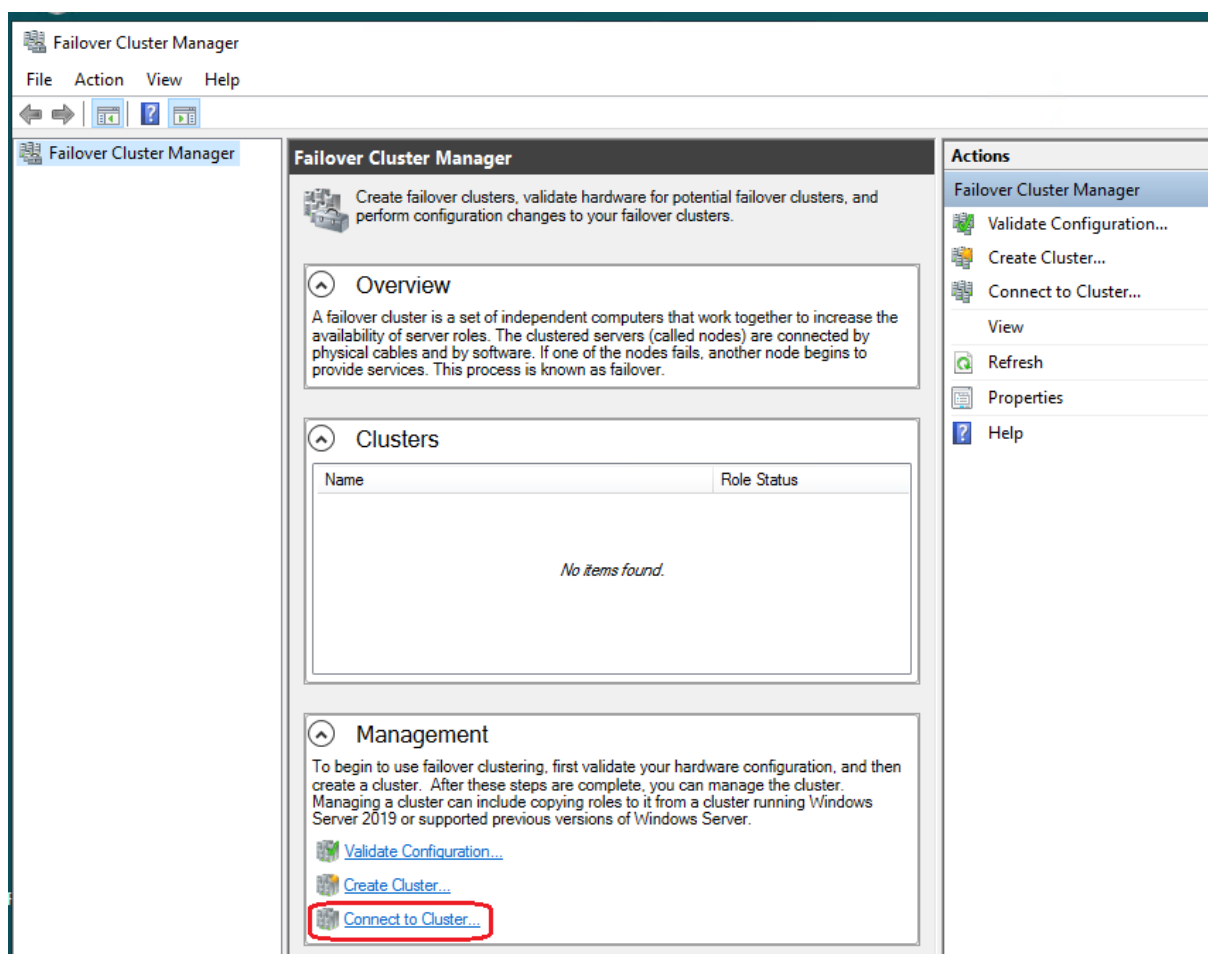
The following sections detail each step:

Identify active node and repository status

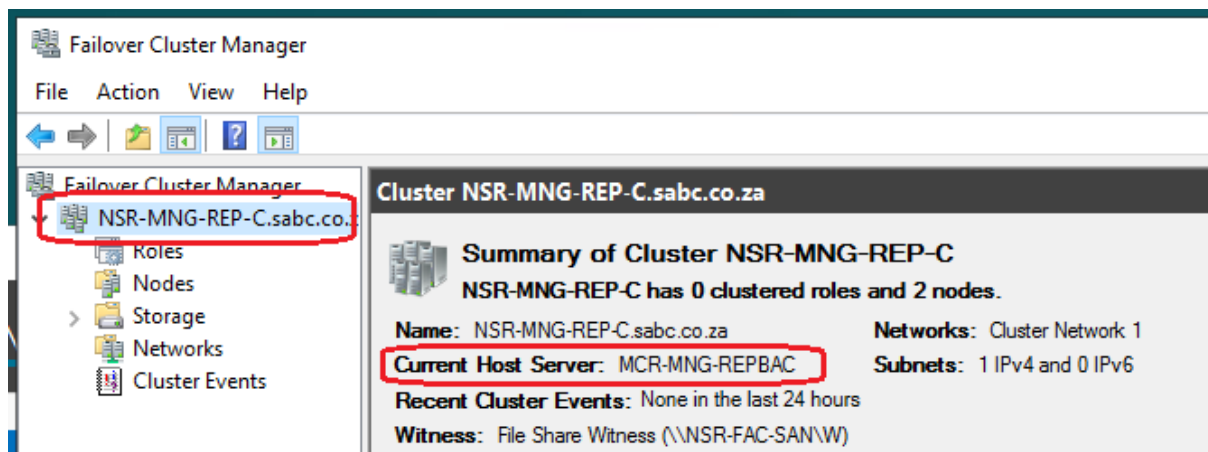
RDP into one of the nodes on the cluster and start the Failover Cluster Manager from the Server Manager menu:



In the Failover Cluster Manager connect to the cluster:



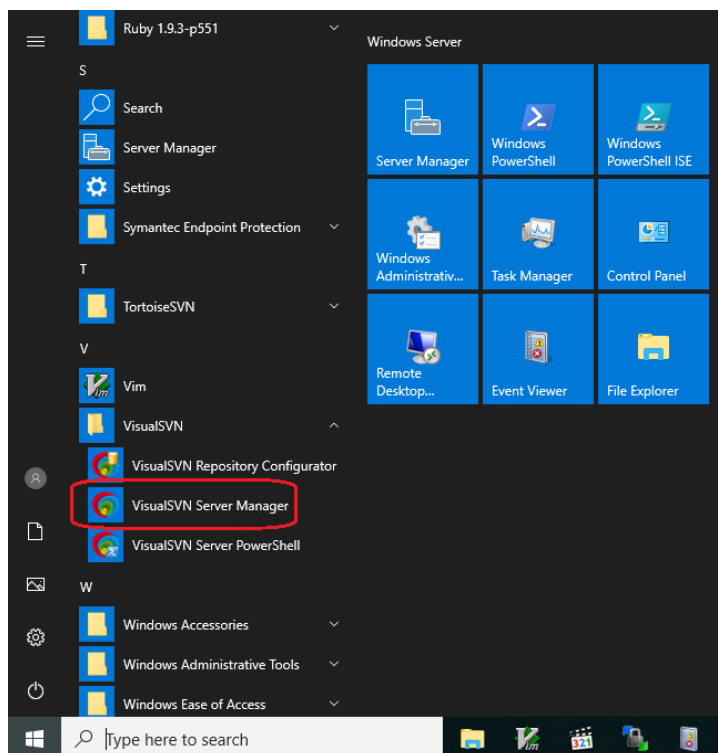
The currently active node is shown at the top of the subsequent menu:



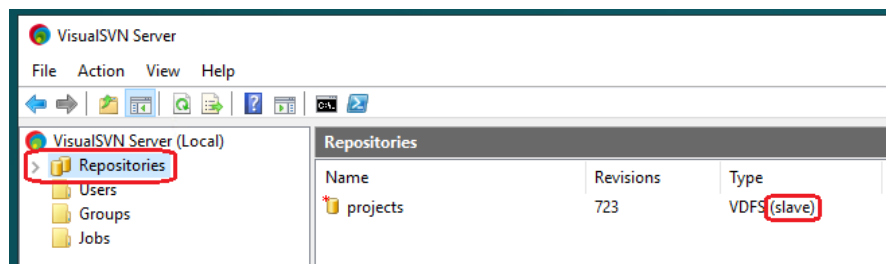
In this example the currently active node in the cluster is server MCR-MNG-REPBAC which is considered to be the “backup” node in the cluster.

Now login to the active cluster node (in this case NSR-MNG-REPBAC) and check the status of the VisualSVN server.

This is done via the VisualSVN Server Manager which is started from the Windows Start icon:



In the VisualSVN Server manager click on the Repositories entry and the status of the “project” repository can be seen on the right of the screen:

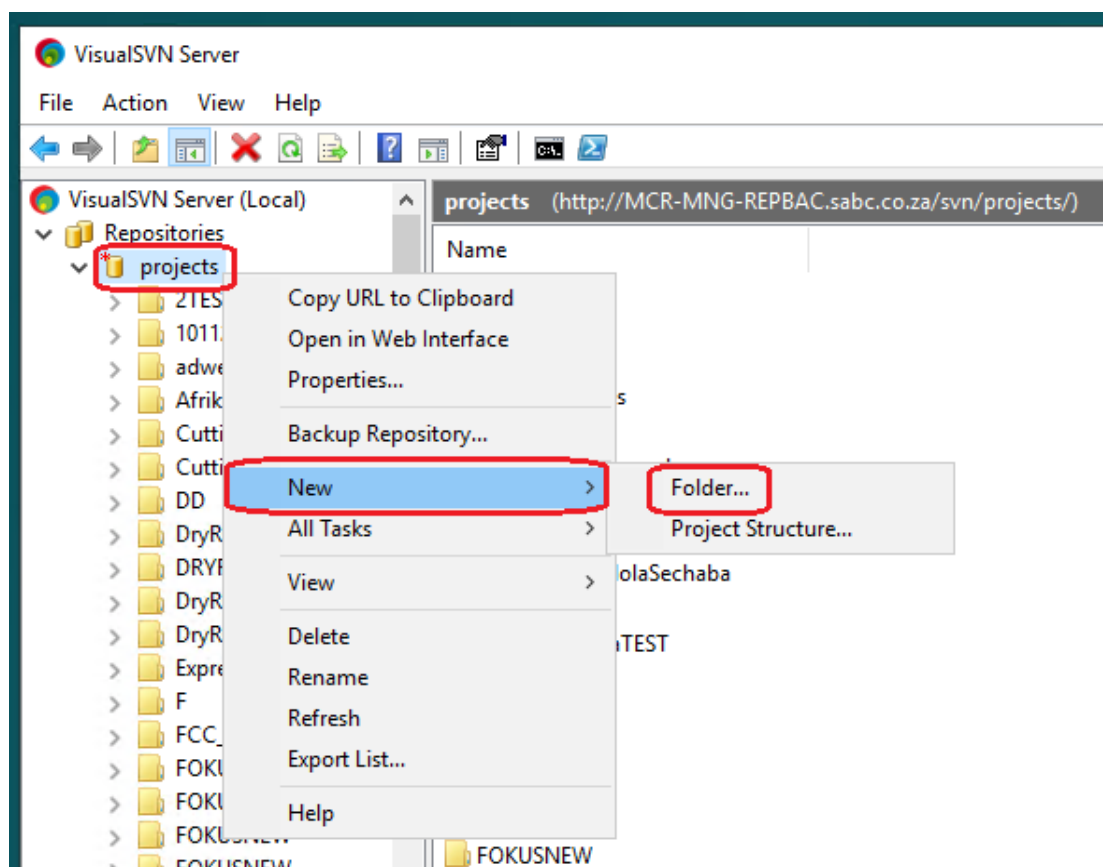


So, in this example we know that the currently active cluster node is the “backup” node and its VisualSVN repository is the “slave” repository.

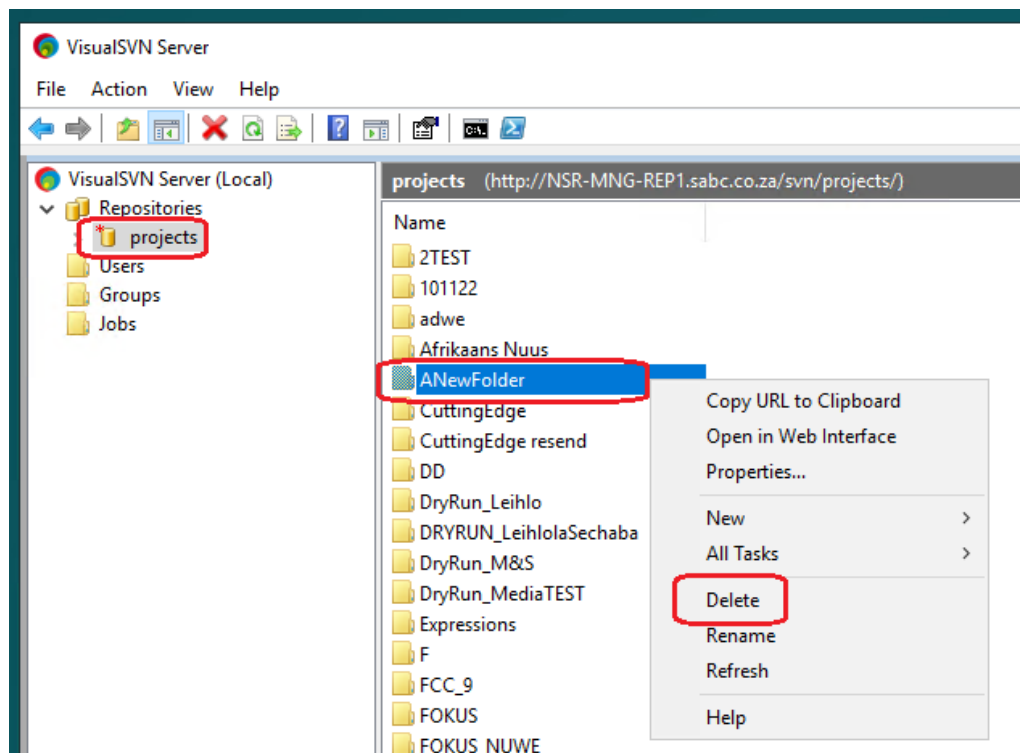
This is what we would expect to see if the “main” node in the cluster had failed and the cluster had failed-over to the “backup” node.

Check if “slave” repository is writable

Using the VisualSVN Repository Manager try and create a new folder in the projects repository. Right-click on the repository name to create a new folder.



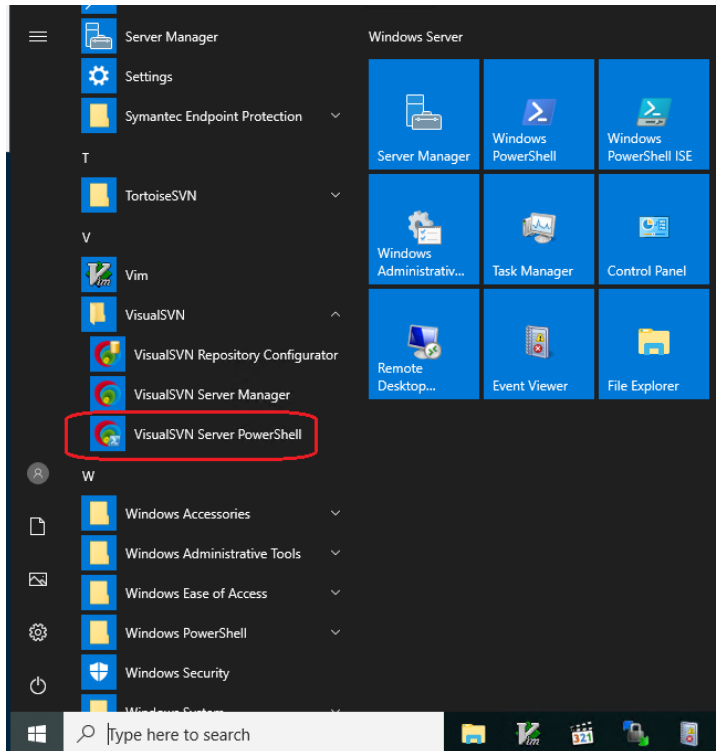
If the folder gets created successfully then the “slave” is writable. Delete the new folder immediately. In this example I created a new folder called “ANewFolder”:



If the folder did not get created then this repository is read-only and the next step is to convert it to a “master” repository to make it read/write.

Convert the “slave” VisualSVN repository to a “master”

Start up the VisualSVN



In the PowerShell window type in the following commands:

Suspend-SvnRepository projects

Switch-SvnRepository -Name projects -Role Master

Set-SvnRepository projects -ReplicationEnabled \$true

Resume-SvnRepository -Name Projects

After running these commands the VisualSVN repository is now a “master” and therefore read/write and the Repository application should now work fine.

At this stage the Repository cluster is being served by the formerly “backup” cluster node and the VisualSVN repository is a read-write “master” repository.

To re-enable the cluster functionality the failed “main” machine in the cluster needs to be repaired and brought back on-line. The VisualSVN repository on the “repaired master” will now need to be made a slave to the new “master” running on the “backup” node in the cluster.

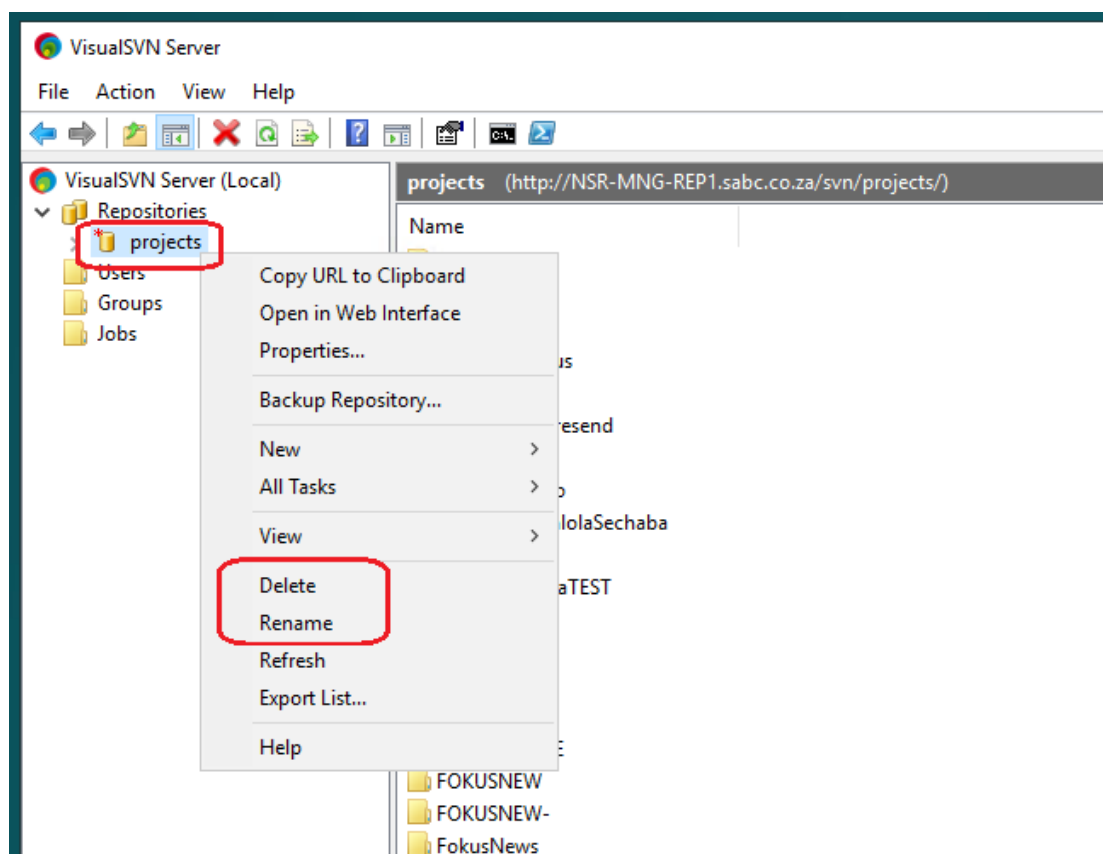
Restore the failed node and re-create the VisualSVN repository as a “slave” to the new “master”

Once the failed “master” node in the cluster is back on-line we need to sync the VisualSVN repository to the new “master” repository.

The first step is to delete or rename the “projects” repository. The second step is to create a new “slave” repository and let it re-sync.

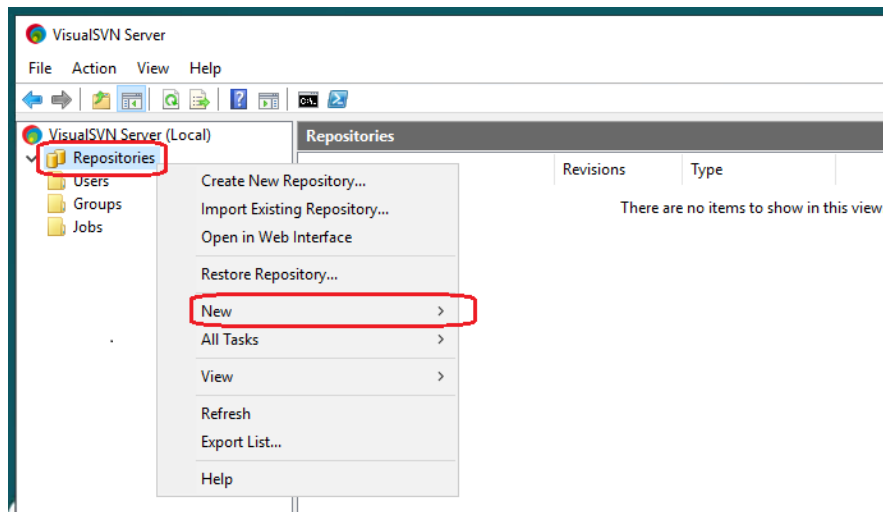
Both of these steps are done via the VisualSVN Server Manager.

To delete or rename the repository called projects right-click on it and select Delete or Rename:

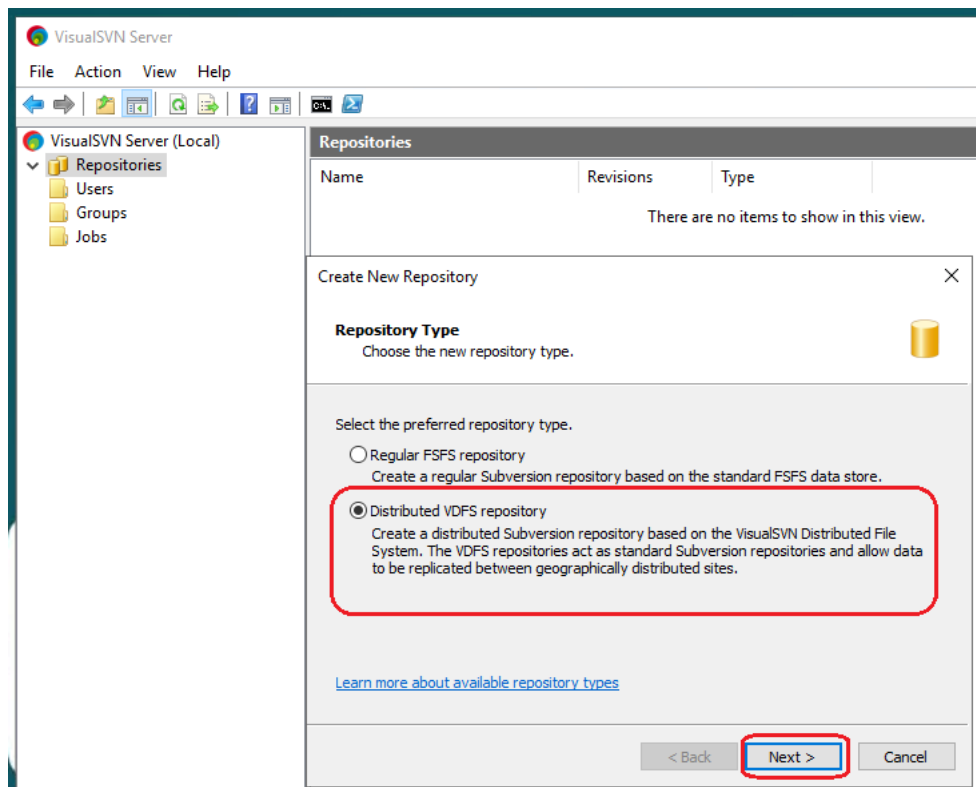


The next step is to create a new repository called projects and make it a “slave” to the new “master” repository.

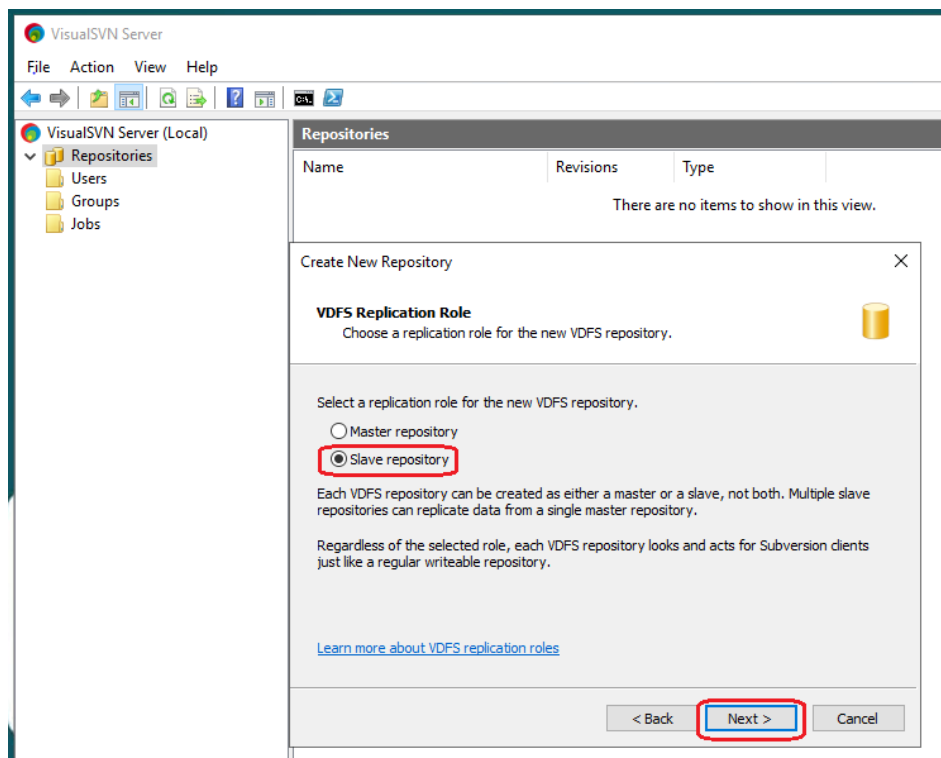
Right-click on “Repositories” and select “New,”



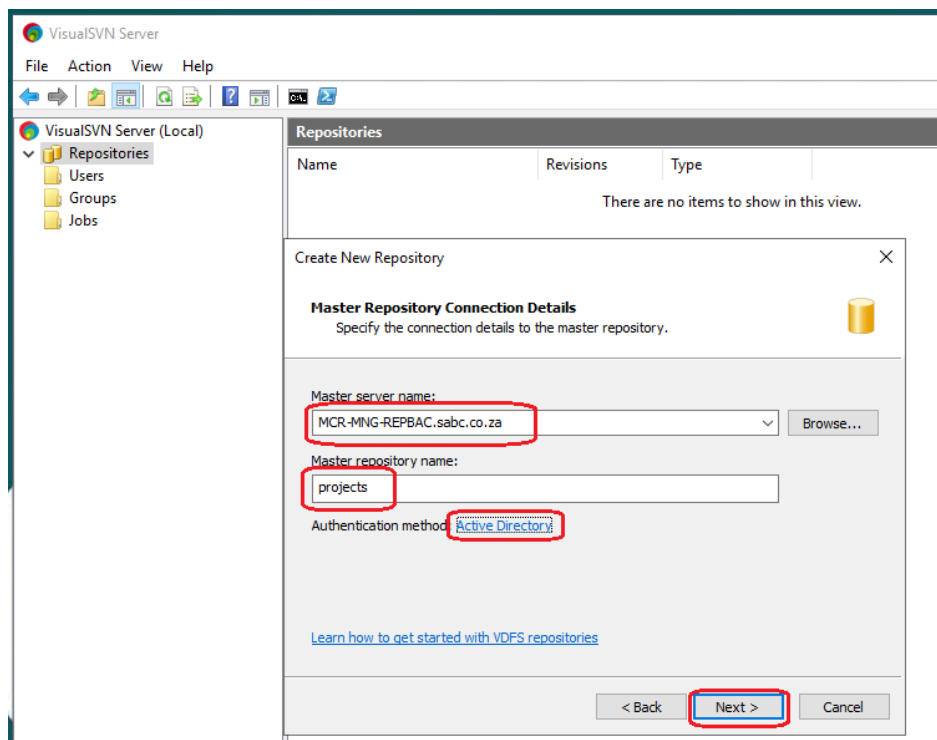
Select it to be a “Distributed VDFS repository”:

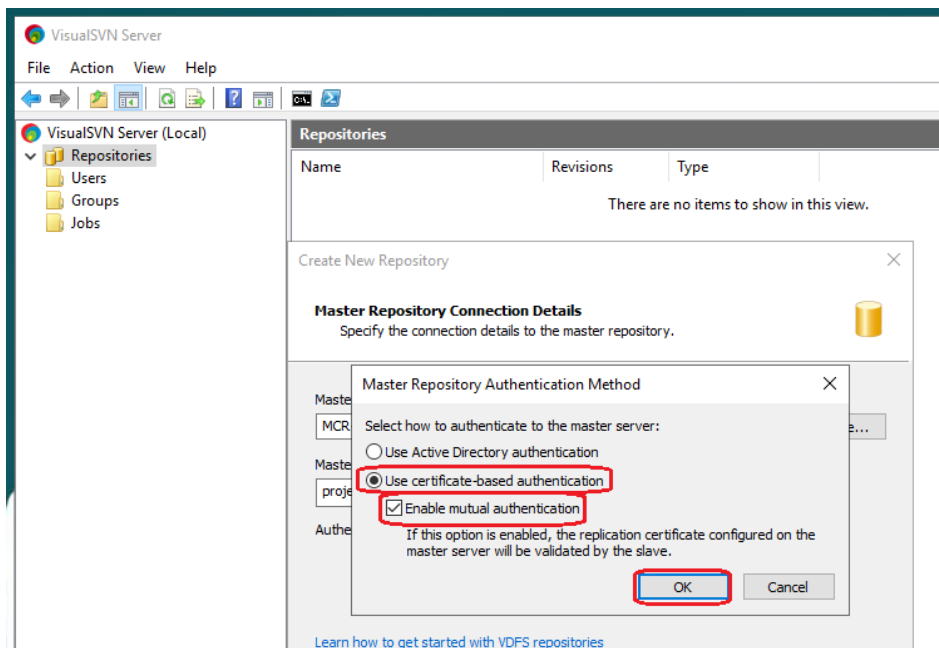


Select it as a “slave” repository:

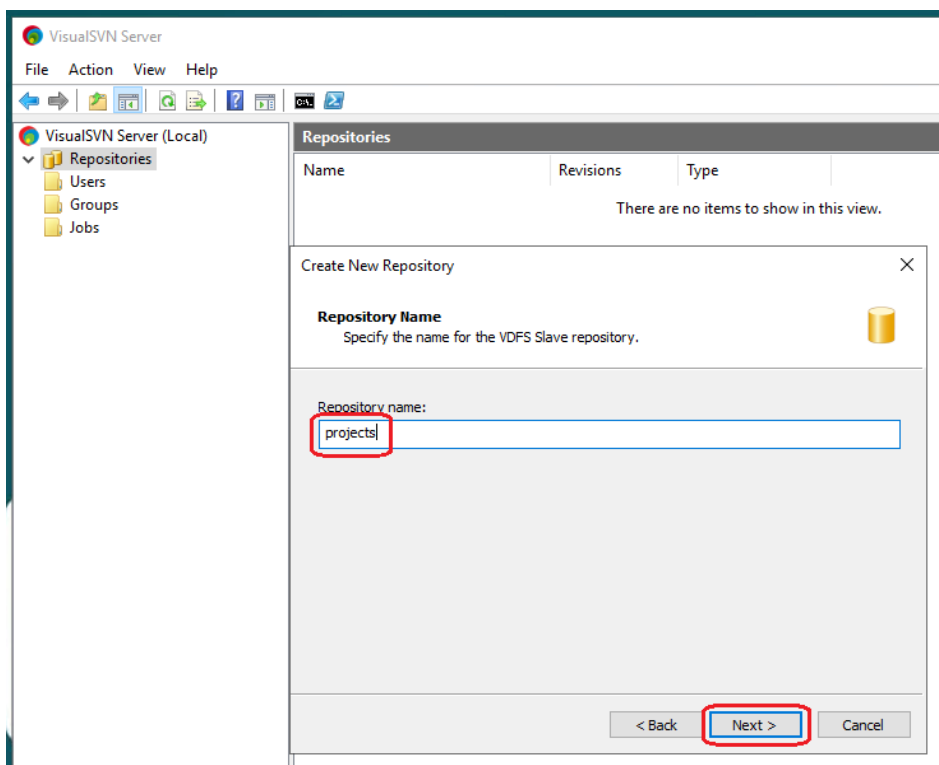


Enter the full name of the active node hosting the new “master” repository and the name of the repository on that node and under “Authentication method” select “Use certificate-based authentication” and tick the “Enable mutual authentication” check-box:

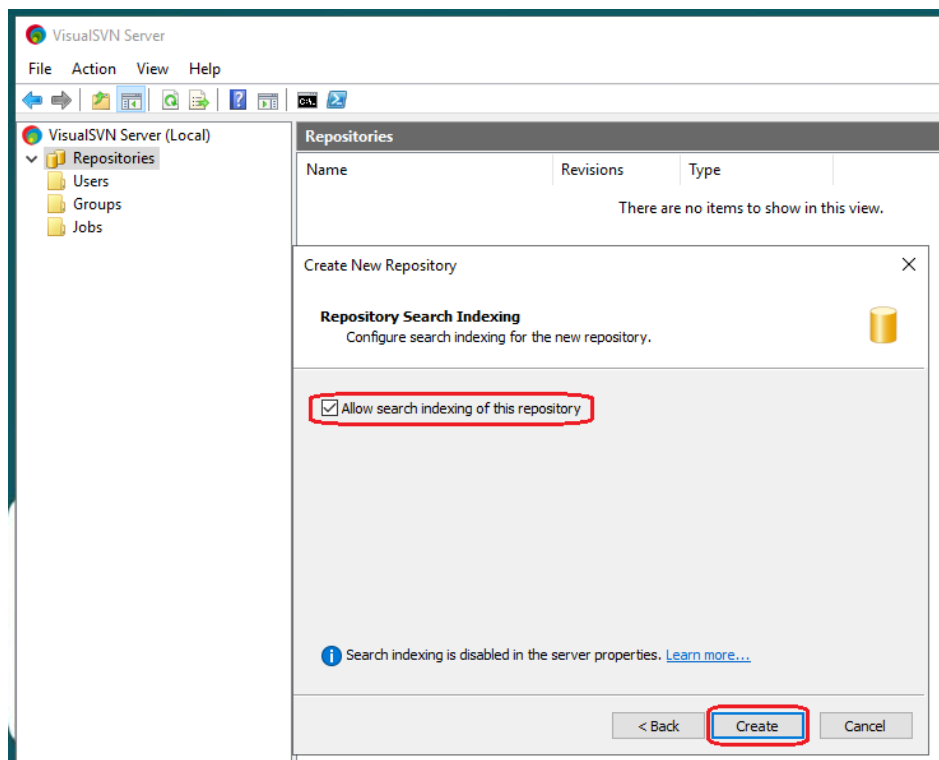




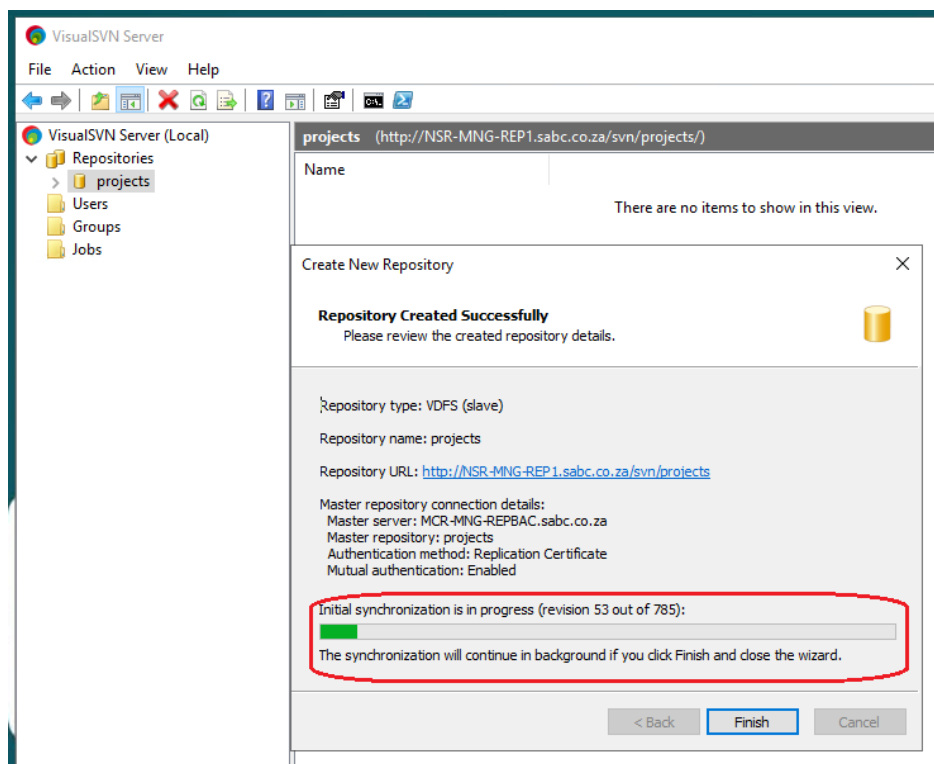
Name the new repository "projects":



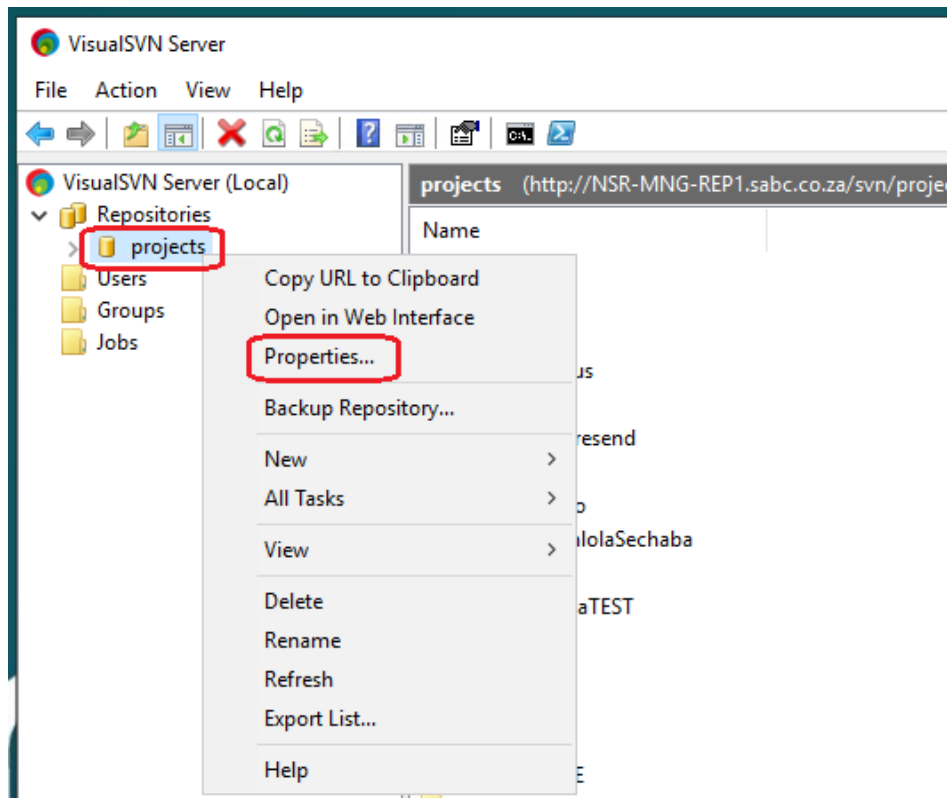
Check “Allow search indexing of this repository” and click “Create”:



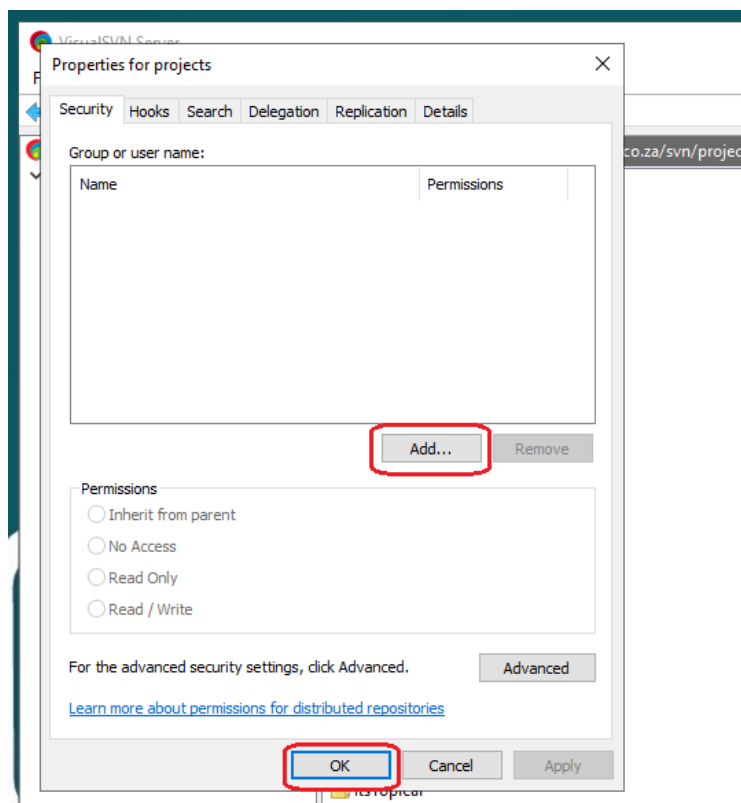
A progress window will now come up. Wait until completion before clicking on “Finish”.



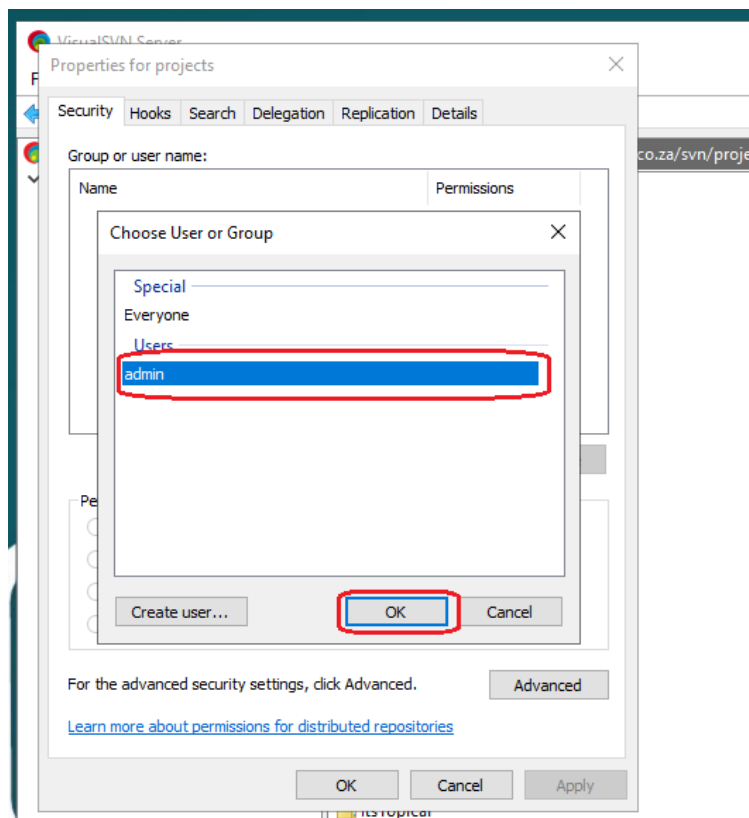
We now need to assign a user permission to access the new repository. Right-click on the repository name and select “Properties...”:



Under the “Security” tab click on “Add...”:



Select user "admin" and click "OK":



A further click on OK and the procedure is finished.

We can now make sure we can access the newly created repository from a cmd window. The user name is "admin" with password "admin":

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.17763.737]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>svn info http://localhost/svn/projects
Path: projects
URL: http://localhost/svn/projects
Relative URL: ^/
Repository Root: http://localhost/svn/projects
Repository UUID: 8f5bd04e-b1e2-a447-a93f-ff099f653665
Revision: 785
Node Kind: directory
Last Changed Author: admin
Last Changed Rev: 785
Last Changed Date: 2023-02-23 17:49:32 +0200 (Thu, 23 Feb 2023)

C:\Users\Administrator>
```