# Solution M4: Clustering and High Availability

There are multiple ways to achieve what was requested

As the visual approach is intuitive, the presented solution is implemented with PowerShell

Please note that the solution is far from being optimal. The aim is to illustrate all (or most) individual steps

## Possible Solution Task 1

Check the **M4-Solution-Script-Clustering-and-High-Availability-Task-1.ps1** file

Adjust it to match your situation

Open a **PowerShell** session with **Run as Administrator** on **the Hyper-V** host and execute it

Here, we will explain just the steps related to the actual solution

Once the domain is up and running and all machines are part of it, we can continue

Add second NIC to both member machines

**Add-VMNetworkAdapter -VMName $VM2, $VM3 -SwitchName "NAT vSwitch" -Passthru | Set-VMNetworkAdapter -MacAddressSpoofing On**

Set the IP address for the second NICs

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { New-NetIPAddress -InterfaceAlias "Ethernet 2" -IPAddress "192.168.99.110" -PrefixLength 24 }**

**Invoke-Command -VMName $VM3 -Credential $DC -ScriptBlock { New-NetIPAddress -InterfaceAlias "Ethernet 2" -IPAddress "192.168.99.111" -PrefixLength 24 }**

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { Set-NetIPInterface -InterfaceAlias "Ethernet 2" -AddressFamily IPv4 -Forwarding Enabled }**

Install NLB feature + IIS Role on all member VMs

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { Install-WindowsFeature NLB, Web-Server -IncludeManagementTools }**

Set a customized web page on each NLB VM

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { Set-Content -Path C:\inetpub\wwwroot\index.html -Value "<h1>Hello world!</h1><br /><br /><i>Served by $(hostname)</i>" -Force }**

Create the NLB cluster

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { New-NlbCluster -InterfaceName "Ethernet 2" -OperationMode Unicast -ClusterPrimaryIP 192.168.99.100 -ClusterName NLBCluster }**

Add the second node

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { Add-NlbClusterNode -InterfaceName "Ethernet 2" -NewNodeName "HW41-SRV2" -NewNodeInterface "Ethernet 2" }**

Adjust the rule

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { Get-NlbClusterPortRule | Set-NlbClusterPortRule -NewProtocol Tcp -NewStartPort 80 -NewEndPort 80 -NewMode Multiple -NewAffinity None }**

Add a DNS record

**Invoke-Command -VMName $VM1 -Credential $DC -ScriptBlock { Add-DNSServerResourceRecordA -ZoneName WSAA.LAB -Name web -Ipv4Address 192.168.99.100 }**

Log on to the DC, open a browser and navigate to **http://web.wsaa.lab** and refresh a few times

## Possible Solution Task 2

Check the **M4-Solution-Script-Clustering-and-High-Availability-Task-2.ps1** file

Adjust it to match your situation

Open a **PowerShell** session with **Run as Administrator** on **the Hyper-V** host and execute it

Here, we will explain just the steps related to the actual solution

Once the domain is up and running and all machines are part of it, we can continue

Add second NIC for storage to all VMs

**Add-VMNetworkAdapter -VMName $VM1, $VM2, $VM3, $VM4 -SwitchName "vStorage"**

Add third NIC for cluster communication to all member servers

**Add-VMNetworkAdapter -VMName $VM2, $VM3, $VM4 -SwitchName "vPrivate"**

Set the IP address for the second NICs

**Invoke-Command -VMName $VM1 -Credential $DC -ScriptBlock { Rename-NetAdapter -Name "Ethernet 2" -NewName "Storage" ; New-NetIPAddress -InterfaceAlias "Storage" -IPAddress "192.168.77.2" -PrefixLength 24 }**

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { Rename-NetAdapter -Name "Ethernet 2" -NewName "Storage" ; New-NetIPAddress -InterfaceAlias "Storage" -IPAddress "192.168.77.10" -PrefixLength 24 }**

**Invoke-Command -VMName $VM3 -Credential $DC -ScriptBlock { Rename-NetAdapter -Name "Ethernet 2" -NewName "Storage" ; New-NetIPAddress -InterfaceAlias "Storage" -IPAddress "192.168.77.11" -PrefixLength 24 }**

**Invoke-Command -VMName $VM4 -Credential $DC -ScriptBlock { Rename-NetAdapter -Name "Ethernet 2" -NewName "Storage" ; New-NetIPAddress -InterfaceAlias "Storage" -IPAddress "192.168.77.12" -PrefixLength 24 }**

Set the IP address for the third NICs

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { Rename-NetAdapter -Name "Ethernet 3" -NewName "Private" ; New-NetIPAddress -InterfaceAlias "Private" -IPAddress "192.168.78.10" -PrefixLength 24 }**

**Invoke-Command -VMName $VM3 -Credential $DC -ScriptBlock { Rename-NetAdapter -Name "Ethernet 3" -NewName "Private" ; New-NetIPAddress -InterfaceAlias "Private" -IPAddress "192.168.78.11" -PrefixLength 24 }**

**Invoke-Command -VMName $VM4 -Credential $DC -ScriptBlock { Rename-NetAdapter -Name "Ethernet 3" -NewName "Private" ; New-NetIPAddress -InterfaceAlias "Private" -IPAddress "192.168.78.12" -PrefixLength 24 }**

Install iSCSI Target

**Invoke-Command -VMName $VM1 -Credential $DC -ScriptBlock { Install-WindowsFeature FS-iSCSITarget-Server }**

Create iSCSI virtual hard disk (quorum)

**Invoke-Command -VMName $VM1 -Credential $DC -ScriptBlock { New-IscsiVirtualDisk -Path "C:\iscsi-disk-quorum.vhdx" -Size 1GB }**

Create iSCSI target (quorum)

**Invoke-Command -VMName $VM1 -Credential $DC -ScriptBlock { New-IscsiServerTarget -TargetName "quorum" -InitiatorId @("IPAddress:192.168.77.10", "IPAddress:192.168.77.11", "IPAddress:192.168.77.12") }**

Attach iSCSI virtual hard disk to an iSCSI target (quorum)

**Invoke-Command -VMName $VM1 -Credential $DC -ScriptBlock { Add-IscsiVirtualDiskTargetMapping -TargetName "quorum" -DevicePath "C:\iscsi-disk-quorum.vhdx" }**

Create iSCSI virtual hard disk (storage)

**Invoke-Command -VMName $VM1 -Credential $DC -ScriptBlock { New-IscsiVirtualDisk -Path "C:\iscsi-disk-storage.vhdx" -Size 5GB }**

Create iSCSI target (storage)

**Invoke-Command -VMName $VM1 -Credential $DC -ScriptBlock { New-IscsiServerTarget -TargetName "storage" -InitiatorId @("IPAddress:192.168.77.10", "IPAddress:192.168.77.11", "IPAddress:192.168.77.12") }**

Attach iSCSI virtual hard disk to an iSCSI target (storage)

**Invoke-Command -VMName $VM1 -Credential $DC -ScriptBlock { Add-IscsiVirtualDiskTargetMapping -TargetName "storage" -DevicePath "C:\iscsi-disk-storage.vhdx" }**

Start iSCSI Initiator service on all member VMs

**Invoke-Command -VMName $VM2, $VM3, $VM4 -Credential $DC -ScriptBlock { Start-Service msiscsi ; Set-Service msiscsi -StartupType Automatic }**

Work out iSCSI targets on member #1. Create new iSCSI target portal

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { New-IscsiTargetPortal -TargetPortalAddress "192.168.77.2" -InitiatorPortalAddress "192.168.77.10" -InitiatorInstanceName "ROOT\ISCSIPRT\0000\_0" }**

Connect to an iSCSI target

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { Get-IscsiTarget | foreach { Connect-IscsiTarget -NodeAddress $\_.NodeAddress -TargetPortalAddress "192.168.77.2" -InitiatorPortalAddress "192.168.77.10" -IsPersistent $true } }**

Initialize and format the disks

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { Initialize-Disk -Number 1 -PartitionStyle GPT ; New-Volume -DiskNumber 1 -FriendlyName "iSCSIDiskQuorum" -FileSystem NTFS -DriveLetter Q }**

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { Initialize-Disk -Number 2 -PartitionStyle GPT ; New-Volume -DiskNumber 2 -FriendlyName "iSCSIDiskStorage" -FileSystem NTFS -DriveLetter S }**

Work out iSCSI targets on member #2. Create new iSCSI target portal

**Invoke-Command -VMName $VM3 -Credential $DC -ScriptBlock { New-IscsiTargetPortal -TargetPortalAddress "192.168.77.2" -InitiatorPortalAddress "192.168.77.11" -InitiatorInstanceName "ROOT\ISCSIPRT\0000\_0" }**

Connect to an iSCSI target

**Invoke-Command -VMName $VM3 -Credential $DC -ScriptBlock { Get-IscsiTarget | foreach { Connect-IscsiTarget -NodeAddress $\_.NodeAddress -TargetPortalAddress "192.168.77.2" -InitiatorPortalAddress "192.168.77.11" -IsPersistent $true } }**

Work out iSCSI targets on member #3. Create new iSCSI target portal

**Invoke-Command -VMName $VM4 -Credential $DC -ScriptBlock { New-IscsiTargetPortal -TargetPortalAddress "192.168.77.2" -InitiatorPortalAddress "192.168.77.12" -InitiatorInstanceName "ROOT\ISCSIPRT\0000\_0" }**

Connect to an iSCSI target

**Invoke-Command -VMName $VM4 -Credential $DC -ScriptBlock { Get-IscsiTarget | foreach { Connect-IscsiTarget -NodeAddress $\_.NodeAddress -TargetPortalAddress "192.168.77.2" -InitiatorPortalAddress "192.168.77.12" -IsPersistent $true } }**

Install failover role + file server role on all member VMs

**Invoke-Command -VMName $VM2, $VM3, $VM4 -Credential $DC -ScriptBlock { Install-WindowsFeature FS-FileServer, Failover-Clustering -IncludeManagementTools -Restart }**

Wait for all member servers to reboot and then continue

Test cluster - optional step. It will take time, so you can skip it

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { Test-Cluster -Node HW42-SRV1, HW42-SRV2, HW42-SRV3 }**

Create the cluster

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { New-Cluster -Name ClusterHW -Node HW42-SRV1, HW42-SRV2, HW42-SRV3 -StaticAddress 192.168.99.33 -NoStorage }**

Add the quorum disk

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { $DQ = Get-ClusterAvailableDisk | Where -Property Size -Eq 1GB ; $DQ | Add-ClusterDisk ; Set-ClusterQuorum -DiskWitness $DQ.Name }**

Add a shared volume to the cluster

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { $DS = Get-ClusterAvailableDisk | Where -Property Size -Eq 5GB ; $DS | Add-ClusterDisk ; Add-ClusterSharedVolume $DS.Name }**

Add scale out file server role

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { Add-ClusterScaleOutFileServerRole }**

Prepare and share the folder

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { New-Item -Path C:\ClusterStorage\Volume1\Shares\DATA -Type Directory -Force }**

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { New-SmbShare -Name "DATA" -Path "C:\ClusterStorage\Volume1\Shares\DATA" -FullAccess Everyone }**

Log on to the first member machine, open **Failover Cluster Manager** and examine the result

## Possible Solution Task 3

Check the **M4-Solution-Script-Clustering-and-High-Availability-Task-3.ps1** file

Adjust it to match your situation

Open a **PowerShell** session with **Run as Administrator** on **the Hyper-V** host and execute it

Here, we will explain just the steps related to the actual solution

Once the domain is up and running and all machines are part of it, we can continue

Role installation – **Containers**, **File Server** and **Hyper-V**

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { Install-WindowsFeature -Name Containers, FS-FileServer, Hyper-V -IncludeManagementTools -Restart }**

Wait for the roles to be installed on the VMs and then continue

Install **Docker Provider** on the member servers

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201 -Force ; Install-Module -Name DockerMsftProvider -Repository PSGallery -Force }**

Install **Docker** on the member servers

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { Install-Package -Name Docker -ProviderName DockerMsftProvider -Force }**

Ensure the **Docker** service is started

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { Start-Service docker }**

Add set of firewall rules to enable correct communication between nodes

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { New-NetFirewallRule -DisplayName "Docker Port 2376/tcp" -Direction Inbound -LocalPort 2376 -Protocol TCP -Action Allow }**

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { New-NetFirewallRule -DisplayName "Docker Port 2377/tcp" -Direction Inbound -LocalPort 2377 -Protocol TCP -Action Allow }**

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { New-NetFirewallRule -DisplayName "Docker Port 4789/udp" -Direction Inbound -LocalPort 4789 -Protocol UDP -Action Allow }**

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { New-NetFirewallRule -DisplayName "Docker Port 7946/tcp" -Direction Inbound -LocalPort 7946 -Protocol TCP -Action Allow }**

**Invoke-Command -VMName $VM2, $VM3 -Credential $DC -ScriptBlock { New-NetFirewallRule -DisplayName "Docker Port 7946/udp" -Direction Inbound -LocalPort 7946 -Protocol UDP -Action Allow }**

Initialize the Swarm on node #1 (SRV1)

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { docker swarm init --advertise-addr 192.168.99.10 ; docker swarm join-token -q worker > c:\swarm-token.txt }**

Join node #2 (SRV2) to the Swarm

**Invoke-Command -VMName $VM3 -Credential $DC -ScriptBlock { docker swarm join --token $(type \\HW43-SRV1\c$\swarm-token.txt) 192.168.99.10:2377 }**

Check the status of the Swarm

**Invoke-Command -VMName $VM2 -Credential $DC -ScriptBlock { docker node ls }**