Lab 4:

Virtualization with   
Microsoft Hyper-V

Datacenter Virtualization

2024-2025

© Daan Pareit

CONTENT

[1 Introduction 3](#_Toc182523608)

[1.1 Learning goals 3](#_Toc182523609)

[1.1.1 Knowledge 3](#_Toc182523610)

[1.1.2 Skills 4](#_Toc182523611)

[1.2 Prerequisites 4](#_Toc182523612)

[2 Installing Windows Server 5](#_Toc182523613)

[2.1 Preface 5](#_Toc182523614)

[2.2 Preparing your OS for nested virtualization 5](#_Toc182523615)

[2.3 Creating the VM to emulate the hypervisor 5](#_Toc182523616)

[2.4 Installation 6](#_Toc182523617)

[2.5 The server name in your hosts file 10](#_Toc182523618)

[3 Remote Server administration 11](#_Toc182523619)

[3.1 Remote CLI: Powershell sessions 11](#_Toc182523620)

[3.2 Remote GUI: Server Manager 18](#_Toc182523621)

[3.3 Remote Web GUI: Windows Admin Center (WAC) 20](#_Toc182523622)

[4 Hyper-V functionality 24](#_Toc182523623)

[4.1 Installing the Hyper-V role 24](#_Toc182523624)

[4.2 Uploading Files to the server 27](#_Toc182523625)

[4.3 Hyper-V Manager 29](#_Toc182523626)

[4.3.1 Installing Hyper-V Manager 29](#_Toc182523627)

[4.3.2 Creating a VM 29](#_Toc182523628)

[4.3.3 Virtual switches 32](#_Toc182523629)

[4.4 PowerShell Hyper-V cmdlets 36](#_Toc182523630)

[4.5 Hyper-V in Windows Admin Center (WAC) 39](#_Toc182523631)

# 

# Introduction

Without Microsoft as a company, many datacenters would cease to exist, albeit because of the database products (SQL Server), the server products (2012R2, 2016, 2019, 2022, 2025) or at the very least the Windows client products that customers expect (Windows 11).

Built into Windows, there is a feature that is quite popular: Microsoft Hyper-V. It is one of the few products to work in the same way on Windows Server as on Windows client platforms. No downloads required, just some checkboxes and one (or two) reboots.

In this lab we’ll explore running VM on Hyper-V in Windows Server 2025.

A close up of a logo

Description automatically generated

## Learning goals

### Knowledge

* Know how Microsoft Server management works
* Know what Hyper-V is and how it works

### Skills

* Be able to install & manage Hyper-V
* Be able to install & manage Hyper-V VMs

## Prerequisites

This lab will be performed on VMware Workstation Pro or VMware Fusion Pro. See earlier lab on how to install this software.

You’ll need the following:

* The 64-bit ISO file for Windows Server 2025. Go to Microsoft Evaluation Center for Windows Server: <https://www.microsoft.com/en-us/evalcenter/download-windows-server-2025>
* The WindowsAdminCenter MSI file. Go to Microsoft Evaluation Center: <https://www.microsoft.com/en-us/evalcenter/evaluate-windows-admin-center>
* To test a low-resource-requiring VM running on Hyper-V, we will boot into a simple Operating System. We’ll use a Tiny Linux distro “TinyCorePure64”, for which the ISO file can be found here: <http://www.tinycorelinux.net/15.x/x86_64/release/TinyCorePure64-15.0.iso>

# Installing Windows Server

## Preface

We will now install a VM with “Windows Server 2025 Datacenter” to serve as the hypervisor for running VMs with Hyper-V. This should only take a few minutes.

## Preparing your OS for nested virtualization

A hypervisor’s purpose is to run virtual machines. But if you install the hypervisor itself within a virtual machine in your VMware Workstation/Fusion, the VMs running on the hypervisor will be VMs running inside your virtual machine. This is what we call **nested virtualization** and the VMs running inside the hypervisor in your VM, are sometimes called **vVMs** (virtual VMs).

If you already have an active virtualization layer on your host OS, running vVMs on your hypervisor Windows Server VM would add two more layers of virtualization, thus three in total. That would be too complex to handle and your VM will fail to run.

This is most notably the case if you are running Windows 11 with Hyper-V enabled on your host (required for e.g. WSL2). In that case: we don’t want you to run VMs on Hyper-V on your own host but on a Windows Server VM with Hyper-V. Of course, the moment you understand how Hyper-V works, nothing stops you from also playing around with Hyper-V on your own host. But for now:

* You need to (temporarily) disable Hyper-V on your Windows 11: see ‘hyperv-vs-vmware.txt’ on Leho

Win -> “features” -> uncheck Hyper-V

bcdedit /set hypervisorlaunchtype off

bcdedit /set hypervisorlaunchtype auto

* If you get the warning that Intel VT-x/EPT is not supported on your platform, it might be due to Credential Guard. It can be disabled using this Microsoft PowerShell script: <https://www.microsoft.com/en-us/download/details.aspx?id=53337>

.\DG\_Readiness\_Tool\_v3.6.ps1 -Disable

## Creating the VM to emulate the hypervisor

Let’s create the VM which will contain your hypervisor:

1. Create a new VM, choose to install a ‘Windows Server 2025’
2. Choose ‘I will install the OS later’ to avoid VMware’s ‘Easy Install’ to kick in.
3. Choose a name and location for the VM files. Preferably this location should not be synchronized with the cloud (onedrive) to avoid instability.
4. Keep the proposed disk (60 GB) and RAM (2GB) size. Store the virtual disk as a single file.
5. Customize hardware to **enable the “Intel VT-x / AMD-V” option**.

**A screenshot of a computer program

Description automatically generated**

**Q: Why do we need this option specifically for this lab?**

**Due to performance issues that may appear, as this allows us to also run “hypervisors”, like we did with ESXi**

Also increase the memory size, preferably to 4GB. Optionally allocate more CPU cores.

## Installation

Now, let’s install Windows Server 2025 on this VM. Instead of using the ‘Desktop Experience’, we’ll rehearse using a ‘server core’ installation instead:

* Use the Windows Server ISO image file in your DVD drive, and boot from that DVD.
* Go for the “Install Windows Server” option to do a clean install
* Choose the installation option “**Datacenter** Evaluation” **without** Desktop Experience.

A screenshot of a computer

Description automatically generated

* Near the end of the installation process, choose a password for Administrator, e.g. ‘Server2025’. (Note: num lock is off by default.)

Password: Server2025

* After logging in for the first time, select the ‘Diagnostic data’ setting and within the ‘sconfig’ tool choose to set Updates to ‘Manual’. Security wise, this is NOT a good idea, but this is no production environment here and we set this option to avoid your VM taking more disk space.

A screenshot of a computer program

Description automatically generated

* To enable your num lock on the login screen, go to the command line and start “regedit.exe”. Then:
  + Go to HKEY\_USERS\.Default\Control Panel\Keyboard
  + Change the value of InitialKeyboardIndicators to 2
* In the VMware Workstation menu, choose to ‘Install VMtools’. In your Windows, this will appear as an inserted CD. On the command line, execute “D:\setup64.exe” and complete the setup. Note that the installer window will typically be hidden behind your terminal window, thus you’ll have to move/minimize that window to see the VM tools installer.
* What is your current server’s hostname? Change it to ‘**hyperhost-<your-firstname>-<your-lastname>**’ This is important to verify your individual assignment! Lab will be graded zero without personalized prompt. You can use the Rename-Computer cmdlet to this end. Confirm the truncated NetBIOS warning and reboot the server to apply the change.

A screenshot of a computer program

Description automatically generated

* To gain back some disk space: shutdown your VM and in VMware workstation choose ‘Manage – Clean up disk space’.
* For your convenience, you could take a snapshot of this cleanly installed VM.

## The server name in your hosts file

Some of the features you’ll consider next rely on host names. As your server is not in an Active Directory, it is recommended here to save the hostname of the Windows Server to the hosts file of your management machine:

* You could edit the hosts file manually to add the relation between the ‘hyperhost-<first>-<last>’ name and its IP address. Or, from an elevated Command Prompt (**not PowerShell**) on our Management Machine, run this command:  
  echo <ip> hyperhost-<first>-<last> >> %SystemRoot%\System32\drivers\etc\hosts
* Verify this by typing   
  ping hyperhost

A screenshot of a computer

Description automatically generated  
🡪 Although the ping itself is failing, we should see the correct IP being pinged, if not open the hosts file, verify it and fix it.  
**Q: Why is there no response to the ping?**

This is probably due to firewall rules that block ICMP by default.

**A screenshot of a computer program

Description automatically generated**

**As we see – most rules are disabled, so here we are not having access to it.  
Hint: let’s rehearse your basic PowerShell skills:**

**Get-NetFirewallRule -name "\*icmp4\*" | ft name,enabled,action,profile**

* As always, make sure you understand the PowerShell cmdlets in the statement above, rather than just copying/executing. Inspect the output and use similar PowerShell cmdlets to make sure ping is working by enabling all firewall rules in the respective displaygroup.

A screen shot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

# Remote Server administration

As we’ve installed Windows Server as a ‘server core’ without desktop experience, we’ll need a **management machine** to manage the Windows Server remotely. Therefore, we need a Windows machine in the same network, this can be e.g. your own laptop Windows host OS or a Windows VM (a ‘Windows 11’ VM or a ‘Windows Server 2025 with Desktop Experience’ VM).

We will now remotely manage our Windows Server hypervisor in different ways: using

1. PowerShell,
2. Server Manager,
3. Windows Admin Center

## Remote CLI: Powershell sessions

Similar to what SSH’ing to a Linux Server is for having a CLI at a remote Linux server, we can use a remote PowerShell session to have a CLI at our remote Windows Server. Let’s start inspecting (or rehearsing) how remote PowerShell works via the ‘Windows Remote Management’ (WinRM).

First, on the Windows Server:

* Step1: configuration?
  + Out of the box a Windows Server is ready to accept incoming ‘Windows Remote Management’ (WinRM) connections. Verify the service is indeed active with the cmdlet ‘Get-Service WinRM’

A screenshot of a computer program

Description automatically generated

* + If the service is running, you can use the “winrm” tool/command. To know how the service is configured, use “winrm get winrm/config/service”.

A screenshot of a computer

Description automatically generated

**Q: What ports are used by default for the WinRM service?**

**HTTP: 5985**

**HTTPS: 5986**

* Step2: active process(es)?
  + That’s for the configuration but is any listener process active? Therefore, use “winrm enum winrm/config/listener”.

A screenshot of a computer

Description automatically generated

**Q: What port(s) is/are listened on?**

**5985, enabled.**

* + Verify that the(se) ports are indeed open with the cmdlet “Get-NetTCPConnection -Localport <port>” or “netstat -ano”

A screen shot of a computer

Description automatically generated

* Step3: firewall?
  + Verify WinRM is allowed by the firewall with: Get-NetFirewallRule -DisplayGroup "Windows Remote Management" | Format-Table Name, Enabled, Direction, Profile, Action

A screenshot of a computer program

Description automatically generated

It is allowed

* + Or to explicitly check the specific port to be allowed according to the firewall rules: Get-NetFirewallRule | Where-Object { $\_.Action -eq "Allow" -and $\_.Enabled -eq "True" } | Get-NetFirewallPortFilter | Where-Object { $\_.LocalPort -eq <port> }

A screen shot of a computer

Description automatically generated

* Step4: verify
  + You can actively test with “Test-NetConnection -ComputerName localhost -Port <port>”

A screenshot of a computer

Description automatically generated

Next, on the management machine (your Windows host OS or a Windows VM):

* You can verify your client config with “winrm get winrm/config/client” and note that (of course) the ports where to connect to match with what we’ve seen for the server. Note -as mentioned before- that winrm commands don’t work if the winrm service is not running. You thus might need to (temporarily) start the service if necessary.

winrm quickconfig

A computer screen shot of a black screen with white text

Description automatically generated

* Out of the box, however, connecting via WinRM to a remote host which is not within your domain is disallowed! The exceptions where you do allow your client to connect to are listed in the “TrustedHosts” key of your client configuration. Add your hyperhost to the TrustedHosts via the following in PowerShell (if in command prompt, omit the single quotes):

winrm set winrm/config/client '@{TrustedHosts="<servername>"}'

A screenshot of a computer

Description automatically generated

Note that there are other ways to query/change the WinRM (TrustedHosts) configuration:

* + WinRM is actually Microsoft’s implementation of the more generic Web Services for Management (WSMan) protocol. This is available via the PSDrive WSMan:\
    - **Get-Item WSMan:\localhost\Client\TrustedHosts**
    - **Set-Item WSMan:\localhost\Client\TrustedHosts -Value "<servername>"**
    - **Clear-Item WSMan:\localhost\Client\TrustedHosts**

**A screen shot of a computer

Description automatically generated**

* + Via regedit.exe, you’ll find the same configurations at:
    - HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\WSMAN\

To summarize: out of the box a Windows Server is configured to accept incoming WinRM connections but on the client side you need to allow outgoing WinRM connections to servers which are not within your domain.

Now, let’s start a remote PS Session:

* You can now use the Enter-PSSession cmdlet at your virtual Windows 11 client to make a remote PowerShell connection for Administrator via:   
  Enter-PSSession -ComputerName hyperhost-<firstname>-<lastname> -Credential (Get-Credential)

A screen shot of a computer

Description automatically generated

* Check on your server with netstat in a command prompt or with Resource Monitor if you can find this remote PowerShell connection.

A screenshot of a computer

Description automatically generated

## Remote GUI: Server Manager

The standalone application ‘Server Manager’ is used a lot for managing a Windows Server and the key software within the ‘Remote Server Administration Tools’ (RSAT). It is a Microsoft application that is running (and started by default) on a Windows Server itself if you would have installed the server *with* Desktop Experience (which we didn’t). It can however also be installed on other machines to manage Windows (core) Servers remotely.

* To install Server Manager, there are multiple options, depending on the Windows version of your management machine (check this via ‘winver.exe’):
  + Your management machine is a Windows Server with Desktop Experience

🡪 no action required, you already have Server Manager

* + Your management machine is a Windows 10/11 Home edition

🡪 easy and fast upgrade to Education edition via free license key for students on academicsoftware.be (or continue on another Windows management machine, e.g. a Windows VM)

* + Your management machine is a Windows 10/11 Education or Pro edition

🡪 Install the “RSAT – Server Manager”. This is available as a ‘Windows optional feature’ (via Settings – Apps – Optional Features).

* Start the Server Manager and right-click ‘All Servers’ and choose ‘Add Server’
* Click on the ‘DNS’ tab and search for the IP address or hostname ‘hyperhost-<firstname>-<lastname>’ and add this server.

I am going to do it from my other Windows Server 2025 machine from previous labs.

A screenshot of a computer

Description automatically generated

* Next click at the left on “All Servers” and notice the appearance of HYPERHOST, although it will probably say something like “Access Denied”.
  + Right click the host and choose “Manage As”, enter the credentials of the host  
    **Q: notice that it will give an error “The name provided is not a properly formed account name”, any idea why?**

Because administrative management is disabled maybe, or that it wants it to act as AD.

* + So, let’s fix it by logging in using the Down-Level Logon Name format with the hostname “hyperhost-<firstname>-<lastname>” as the “domain” (to refer to a local user rather than an AD user) and have your Windows Server online in your Server Manager

To solve the issue – Go and install **RSAT: Server Manager** on your host, and then try to add it there. For some reason form another Windows Server it just does not want to work, so I had to do it like this and it worked.

Also the server should be in the TrustedHosts of WinRM with the

To login as a user:

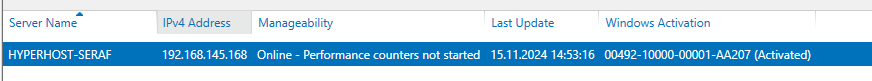
hypersoft-firstname-lastnam\Administrator

password

Or

Administrator@hypersoft-firstname-lastnam

password



## Remote Web GUI: Windows Admin Center (WAC)

Microsoft allows administrators to manage their servers from a central location by providing them with a set of remote administration tools like Server Manager and RSAT/MMC plugins. These, however, typically lack integration, which means that the administrator needs to use a different tool, depending on the settings (s)he wants to modify. To make the administrators’ lives easier, MS has introduced the “Windows Admin Center” (WAC) software in 2018.

WAC allows to remotely administer e.g. Active Directory (AD) domains, Azure clusters, Windows Server (2012 and above) and Client systems and Hyper-V VMs, and it integrates functionalities that are otherwise distributed over many different administration tools.

The Windows Admin Center software can be installed on Windows Server (2016 and above) or on Windows 10/11 machines. However, for performance and security reasons, Microsoft does not allow installation of this software on an Active Directory (AD) Domain Controller (DC). WAC therefore must be installed on a non-DC (managed) gateway server or on the local client system from where the administrator wants to manage the server(s).

* Run the Windows Admin Center MSI file (which you’ve downloaded before) on your Windows management machine. Keep the install settings as they are proposed/recommended in the installer.
* Restart your Windows management machine if necessary, and start Windows Admin Center. Select to use the “Windows Admin Center Client” certificate, as was recommended in the installer as the final “one more thing…” message. If you get an error the first time (“something went wrong”), close the browser window and restart Windows Admin Center.
* In the start window of WAC (all connections), click the “+Add” button and select to add a server. As server name, enter your hyperhost-<first>-<last> Windows Server.

A screenshot of a computer

Description automatically generated

* Now the server should appear in the list of managed devices. Select the checkbox to the left of the servername and click “manage as”. Use the “administrator” account to connect to the server. (you’ll need to set this user again every time you restart WAC)
* Finally, connect to the server by clicking the server name.   
  (The first time, you may get a connection error, but this should be resolved by closing the browser window and restarting Windows Admin Center. )

A screenshot of a computer

Description automatically generated

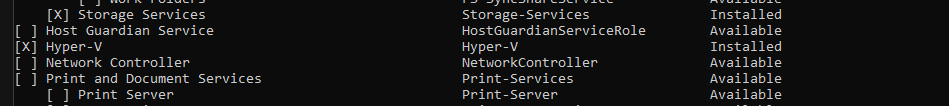
* Look around in WAC and notice how you can control your server from within a web browser!

# Hyper-V functionality

We now have a Windows Server which we can manage remotely in multiple ways. The server can be used for different purposes (domain controller, storage spaces etc.) but in this lab we want to use it as a hypervisor to run VMs.

## Installing the Hyper-V role

* First, check if the Hyper-V feature is already installed out of the box. Check all features with the cmdlet: ‘Get-WindowsFeature’
* To install the ‘Hyper-V’ role, you can right click on the server in your Server Manager and choose to install ‘Roles and Features’. Choose to install the ‘Hyper-V’ role and accept all default values and proposed related features. You’ll get a warning about hardware prerequisites which cannot be validated. This is because we’re running Hyper-V within a VM. You can continue as you’ve manually made sure the VM meets ‘hardware’ requirements by setting the Intel VT-x / AMD-V option for nested virtualization.



**Q: After installing the role, the Windows Server needs to restart. Why is that definitely necessary here?**

**After installing a hypervisor, the windows OS will detach from hardware and put the hyper-v in between.**

## Uploading Files to the server

On a side note: when we’ll create VMs, we’ll typically need to boot/install from ISO files which we’ll also need to get onto the server in this case. So, how to put files on our remote server?

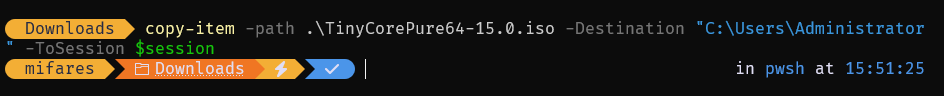
Via Powershell:

* Let’s see how to copy files via remote Powershell, as you would do with ‘scp’ on linux. Exit the remote powershell and now create a session without having an immediate interactive PS shell. Therefore, we’ll use ‘New-PSSession’ instead of ‘Enter-PSSession’:

$session=New-PSSession -ComputerName hyperhost-<firstname>-<lastname> -Credential (Get-Credential)

* You can then copy a file easily via:

Copy-Item -Path "C:\path\to\your\file.txt" -Destination "C:\destination\path\on\remote\server\" -ToSession $session



* You can still use that session as well interactively via:

Enter-PSSession -Session $session

**A screenshot of a computer screen

Description automatically generated**

* When you’re finished, you can clean up your session:

Remove-PSSession $session

Via SMB file shares:

* On the Windows Server, find out what file shares exist with: Get-smbshare . You’ll see the hidden “c$” share.

A screenshot of a computer

Description automatically generated

* Now on your management Windows PC, try to “dir \\hyperhost-<firstname>-<lastname>\c$”. You’ll get a “The user name or password is incorrect”
* We’ll fix that by providing and saving credentials with the “cmdkey” command. This is easily done via: cmdkey /add:hyperhost-<firstname>-<lastname> /user:administrator /pass
* Now, the “dir \\hyperhost-<firstname>-<lastname>” should be successful.

A black screen with white text

Description automatically generated

This does not work but whatever

* Next, you can copy files via:

Copy-Item -Path "C:\path\to\your\file.txt" -Destination "\\hyperhost-<firstname>-<lastname>\c$\"

* Or, in a GUI: just type \\hyperhost-<firstname>-<lastname>\c$ in your Windows start menu or Run window.

Now, upload the ISO file we’ll use:

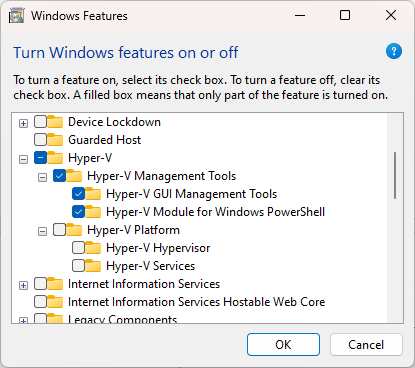
* Copy the ISO file of the Tiny Linux distro to the hyperhost via the method you prefer in a directory of your choice.

## Hyper-V Manager

Now the real deal: using Hyper-V, let’s create some VM’s and configure some settings.

### Installing Hyper-V Manager

* First, install **Hyper-V Management Tools** (not the Hyper-V platform itself) on your **Management Machine**. This is a ‘Windows feature’ (note: different from ‘Optional Feature’ in Windows which you needed to install the RSAT Server Manager).

  
*Hyper-V Management Tools installation on your Management Machine*

* Open the Hyper-V Manager and connect to your ‘hyperhost’ Windows Server

A screenshot of a computer

Description automatically generated

### Creating a VM

* Go to New > Virtual Machine
  1. The name should be Tiny-<firstname>-<lastname>, default location is OK
  2. Generation 1 is required for this Operating System
  3. For memory, 512MB is enough
  4. Notice that we are not able to select any Network Connection  
     **Q: Why? (We will fix this later)**

It might not have the information, or maybe it does not have drivers for the NICs and everything like that.

* 1. The Virtual Hard drive is also not important (we will not install anything), just click next.
  2. Finish and ‘Connect’ to it (via right mouse click or double click).
* Start the (v)VM. What happens? Why?

We do not have a boot device that it can work from

A screenshot of a computer

Description automatically generated

* Now, shut down the (v)VM and open its settings. Configure the CD/DVD drive to use the ISO of the Tiny Linux distro and boot the (v)VM again

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* If you are inside the VM and auto mouse release would not work, then the key combo to release your mouse is by default in Hyper-V: “ctrl + alt + left arrow”. Just so you know 😊.
* Have a look at the VM properties.

**Q: What file type is the Hard Drive? (It makes sense within the Microsoft ecosystem as in previous labs)**

**Vhdx (correct answer for quiz 😊 )**

**Q: Where is it stored?**

**C:\ProgramData\Microsoft\Windows\Virtual Hard Disks**

* Have a look at the overall properties of the Hyperhost itself as well.

**Q: Where are all the VM configuration files stored?**

**C:\ProgramData\Microsoft\Windows\Hyper-V\Virtual Machines**

**A screenshot of a computer

Description automatically generated**

**Q: What can you tell about the difference in storage layout for VMs between Hyper-V and VMware?**

**Hyper-V puts everything in one place I think, but VMware has different directories.**

### Virtual switches

* Use the mouse (or press Alt-F1) to open the menu and find a root console within the Tiny Core OS. If the font size is too small, start a new console with: ‘aterm -fn 10x20’ or another dimension.
* Get the network configuration of the VM via the legacy command ‘ifconfig’ (the regular ‘ip’ command is not yet on Tiny Linux). Remember that during the creation of the VM, we were not able to select any Network Connection. Hence, we don’t have an IP address on the ethernet interface.

A computer screen shot of a blue screen

Description automatically generated

We do want our (v)VM to be connected to the Internet. Therefore, we’ll have to create a ‘Virtual Switch’ in Hyper-V.

* Open the Virtual Switch Manager  
  **Q: What is the difference between "Internal" & "Private"?**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Private – does not allow internet

Internal – allows internet

* Create an External Switch, use a good name (like “Internet” 😉)

A screenshot of a computer

Description automatically generated  
🡪 Make sure that “Allow management operating system to share this network adapter” is turned on. This is extremely important!  
**Q: Why is this extremely important? What does that mean?**

**Then it means that the machine which we are right now on, the hypervisor, can share the network adapter. To me that sounds about right, since even in VMware we have the VMnet adapters.**

* Notice that once we click Apply or OK, the network gets disrupted (which is due to the answer to the previous question).
* Once the virtual switch is created, open the settings to the Tiny Core VM again and reconfigure its network adapter to use the newly created Virtual Switch.

A screenshot of a computer

Description automatically generated

* Now check the network configuration again and verify that the VM has obtained an IP address. Verify the VM has Internet connectivity (e.g. pinging a remote server).  
  **Q: Has a NAT network (as in your Workstation/Fusion) been created by the Virtual Switch? Or is it bridged mode where your VMs are on the same network as your hypervisor?**

It looks like the virtual machine is in bridged mode, since the IP address differs only in terms of the last 8 bits, and the netmask says the same.

A computer screen shot of a blue screen

Description automatically generated

**Q: Take a screenshot of your Hyper-V Manager with your Server connected and your Virtual Machine running with successful connectivity**

A screenshot of a computer

Description automatically generated

## PowerShell Hyper-V cmdlets

When installing the Hyper-V management tools in the Windows Features, you’ve also selected a dedicated PowerShell module to be installed.

* With “get-module”, you can see what imported modules are on your management machine and on your Windows Server. Which one was added when you’ve enabled the Windows Feature for Hyper-V management?

A screen shot of a computer

Description automatically generated

I believe it is the WSMan one, since there is really nothing else that looks added and can be similar to Hyper-V

* With “get-command -Module <Module name>” you see all cmdlets which are now additionally available.
* Execute in your remote PS session to or in your Windows Server locally the “Get-VM” cmdlet when your Tiny VM is running.

A screen shot of a computer program

Description automatically generated

* Now, within a local PowerShell on your management machine, you can also retrieve that information remotely without first entering a remote PS session. Execute   
  “Get-VM -Computername hyperhost-<first>-<last>”   
  **Q: Take a screenshot of this command and its output on your management machine when your Tiny VM is running. Ensure it is showing your personal prompt.**

A screenshot of a computer program

Description automatically generated

* Also starting, pausing and stopping is possible with ‘Start-VM’, ‘Suspend-VM’ and ‘Stop-VM’, again immediately from your management machine without interactive remote powershell. E.g., stop your running Tiny:

Stop-VM -ComputerName hyperhost-<first>-<last> -name tiny-<first>-<last>

* You can also list the different Virtual Switches with ‘Get-VMSwitch’ and a lot more via PowerShell cmdlets.

A screenshot of a computer program

Description automatically generated

## Hyper-V in Windows Admin Center (WAC)

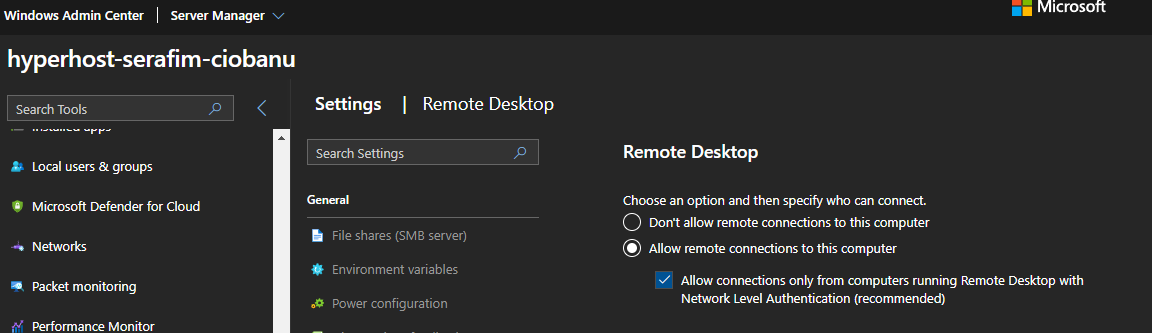
Also in Windows Admin Center (WAC), we can manage the VMs:

* Open WAC and go to your hyperhost server (don’t forget first to ‘manage as’ administrator first).
* In the left side panel, you can specifically choose Virtual Machines’
* Click your tiny-<first>-<last> VM to see its settings. If it’s not running, you can power it on from this web interface.

A black screen with many small colored lines

Description automatically generated with medium confidence

* Choose to ‘Connect’ to the VM. You’ll be redirected to the Settings to enable Remote Desktop first which is OK to do so here.



* When connecting, provide the credentials of your Windows Server. Next, you’ll have your VM GUI available within your browser!

**Q: Take a screenshot of your VM running in your browser via WAC**

**A screenshot of a computer

Description automatically generated**

* Alternatively, you can also choose at the Connect drop-down menu to download a RDP file. Do this and then run the RDP file. You now have a dedicated window for your VM.
* Alternatively, you can also choose from the Connect drop-down menu to download a RDP file. Do this and then run the RDP file. You now have a dedicated window for your VM.

A screenshot of a computer

Description automatically generated

* Also in WAC, you can choose ‘Virtual Switches’ in the left side panel to create/delete/manage your virtual networks.