Lab 12:

Windows installation

Windows Server Security  
 2024-2025

© Daan Pareit

## Introduction

# Lab concept

In these exercises we recapitulate some of the basic skills of Windows (Server) installation/configuration to get our Windows network up and running. We’ll create two new VMs running a Windows (Server) OS.



# Learning goals

* Installing Windows Server Datacenter edition, Core installation
* Installing Windows 11 Enterprise
* Windows network configuration
* Installing and using Remote Management tools (PowerShell Remoting, Server Manager, Windows Admin Center)

# Practicalities and prerequisites

You’ll need the following:

* About 33 GB of free disk space ☹
  + 11 GB for downloaded iso files (6 GB Windows Server 2025, 5 GB Windows 11)
  + 22 GB for fresh installed Windows VMs (8 GB Windows Server 2025, 14 GB Windows 11)
* Go to Microsoft Evaluation Center for Windows Server 2025:   
  https://info.microsoft.com/ww-landing-evaluate-windows-server-2025.html
  + The 64-bit ISO file for Windows Server 2025.
* Go to Microsoft Evaluation Center for Windows Admin Center:  
   <https://info.microsoft.com/ww-landing-windows-admin-center.html>
  + Download the MSI file for Windows Admin Center
* Go to Microsoft Evaluation Center for Windows 11:  
   <https://info.microsoft.com/ww-landing-windows-11-enterprise.html>
  + Download the ISO file for Windows 11 Enterprise
* Make sure your pfSense is running for Internet connectivity for your new VMs.

## Install a Windows Server Datacenter ‘Core’

We’ll install a Windows Server Datacenter ‘core’ edition. A core server is a Windows server that only includes a limited set of GUI features, and requires less hardware resources than editions that include the full desktop experience. In practice, however, it supports almost all roles and features as servers with full desktop experience support. Due to its smaller codebase, it is considered more stable and secure, and it is typically administered using the command line, PowerShell or a remote administration solution (more on that later). For Server 2025, the main minimum hardware requirements are:

* + A 1.4 GHz 64-bit processor, supporting x64, NX, DEP, CMPXCHG16b, LAHF/SAHF, and PrefetchW and Second Level Address Translation (EPT or NPT)
  + 512MB RAM (more if used as a domain controller)
  + A 32 GB hard drive
  + A network adapter

Since we want to use the server machine as a Domain Controller, we’ll create a Virtual Machine in VMware that has somewhat better specifications than the bare minimum:

* Create a new VM, choose to do a typical install and select to ‘install the OS later’ to avoid VMware’s ‘Easy Install’ to kick in.
* Install a ‘Windows Server 2025’
* Choose a name and location for the VM files. Preferably this location should not be synchronized with the cloud (onedrive) to avoid instability.
* Keep the proposed disk (60 GB) and RAM (2GB) size. Store the virtual disk as a single file.
* Set your network adapter to the vmnet where your pfSense and your linux machines are attached to (the one with subnet 192.168.11.0/24).
* 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. What is the difference between the other options?

At least the other ones have more functionality and of course desktop experience.

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

PASSWORD SET TO “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.
* To reclaim some disk space: shutdown your VM and in VMware workstation choose ‘Manage – Clean up disk space’.

## Install a Windows 11 client

We’ll also install a Windows 11 client. For Windows 11 the following are the minimum hardware requirements:

* + A 1.0GHz 64-bit processor with 2 or more cores
  + 4GB RAM
  + 64GB hard disk or SSD
  + A Trusted Platform Module (TPM), compliant to v2.0 of the TPM specifications (more on this later)
  + UEFI and secure boot (need to be available, must be active for e.g. Bitlocker)
* Create a new VM, choose to install a ‘Windows 11 **x64**’. Note: it’s important to choose that option, to get the correct preconfigured hardware for the next labs as well.
* Windows 11 requires a Trusted Platform Module (TPM) piece of hardware, which is automatically (virtually) added by VMware in the wizard. The VM itself will also be obligatory encrypted. Choose (and remember) a password for encryption.

Password = “Windows11”

* Verify in Options-Advanced that your VM has UEFI firmware (instead of BIOS).
* Set your network adapter to the VMnet where your pfSense and your linux machines are attached to (the one with subnet 192.168.11.0/24).
* Use the Windows 11 ISO image file in your DVD drive, to boot from.
* Go for the “Install Windows 11” option to do a clean install
* It will ask you for a Microsoft Account. We don’t want that here (but Microsoft is pushing you that way). To install with a regular offline account, you’ll need to choose “Sign-in options” and then “Domain join instead”.
* Create an account ‘Mickey Mouse’ with password ‘Mouse’.
* You can decline all proposed options (location, personalized ads, etc.)

A screenshot of a computer update

Description automatically generated

WELL THIS IS SOMETHING NEW

Now you have installed your Windows 11 Client, but you need to perform a few tweaks:

* In VMware Workstation, choose to ‘install VMtools’. In your Windows, this will appear as an inserted CD. Go via GUI or Windows Terminal to the CD (D: drive) and start the setup64.exe .
* Set Windows Updates off via gpedit.msc and go to ‘Administrative templates – Windows Components – Windows Update – Manage end user experience – Configure Automatic Updates – disable). Concerning security, this is NOT a good idea, but this is no production environment here and we set this option to avoid your VM taking up more disk space.
* Additionally, open services.msc and go to the ‘Windows Update’ service. Stop the running service and change the ‘Startup type’ to ‘Disabled’.
* To enable your num lock on the login screen, start regedit.exe and :
  + Got to HKEY\_USERS\.Default\Control Panel\Keyboard
  + Change the value of ‘InitialKeyboardIndicators’ to 2
* 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.

## Comparison

Now let’s take a moment to inspect and compare the different Windows VMs you’ve created.

* Execute the following and compare for the different Windows VMs:  
  Get-ItemProperty -Path "HKLM:\Software\Microsoft\Windows NT\CurrentVersion" | Select ProductName, ReleaseId, InstallationType, CurrentMajorVersionNumber,CurrentMinorVersionNumber,CurrentBuild

What kernel version does Windows Server 2025 have (major.minor)? And which one does Windows 11 have?

A screen shot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

For both of them it looks like 10.0

* Compare the size of the virtual hard disk after these clean installs. Which one is larger?





## Some basic local configurations at Windows Server

To configure the ‘core’ server, we do not have a ‘Server Manager’ GUI locally available at the server itself. We will now use PowerShell to configure the basics:

* Login to the Windows Server, as ‘Administrator’
* Exit the sconfig to return to powershell.
* Change the hostname of the Windows Server to ‘KING’. This can be done via sconfig or by using a Powershell cmdlet. If using PowerShell, which cmdlet to use? General quick method: you can use the Get-Command cmdlet to e.g. find cmdlets which have \*name\* as part of the cmdlet. You’ll quickly identify the appropriate cmdlet then. Restart the server afterwards.

Note: You can use the Get-Help cmdlet to obtain more info.

**Note: Ctrl-Space is very useful for autocompleting cmdlets and its options**

**Rename-computer -NewName “KING”**

**Restart-Computer**

* We’ll also have to configure a fixed IP address for the ‘core server’. This can be done via sconfig (easiest way) or PowerShell. Set it to **192.168.11.50/24** and set the pfSense as gateway and DNS server.
* To check if the IP configuration is successful, you can ping from your laptop (192.168.11.1) to the server (192.168.11.50). However, incoming ping requests are blocked by default at the server. Therefore, you’ll have to enable the appropriate Inbound Rule in Windows Firewall. We need to find the name of the firewall rule to enable. Use the following command to list the rules which contain ‘ICMP’ in their name. Do you understand how this command is built?

Get-NetFirewallRule -DisplayName \*ICMP\* | where Enabled -like "False" | select displayname,name,enabled

Get the rules that have ICMP in it, which are disabled, and then select only this fields for it.

Now, you see that the name of the rule we want to enable is ‘FPS-ICMP4-ERQ-In’ (or CoreNet-Diag-ICMP4-EchoRequest-In would work as well)

A computer screen shot of a black screen

Description automatically generated

* With what cmdlet can you enable the ‘FPS-ICMP4-ERQ-In’ firewall rule?

Enable-NetFirewallRule -Name “FPS-ICMP4-ERQ-In”

* Ping the core server from your laptop and verify it is successful. (You might want to enable that rule on your laptop’s Windows Firewall as well for a successful ping. Via the same cmdlets or in the Windows Firewall GUI in the ‘File and Printer Sharing’ group.)

A screenshot of a computer

Description automatically generated

* Now, use New-LocalUser to create a new user “John Doe”. The password needs to conform to the complexity requirements (e.g. Howest123!). Next, use Add-LocalGroupMember to add him to the ‘Users’ group (otherwise he won’t be able to login).

New-LocalUser -Name "John Doe" -Description "Lab 12 user"

A screenshot of a computer

Description automatically generated  
Add-LocalGroupMember -Group "Users" -Member "John Doe"

* Log out as Administrator (via sconfig or logoff.exe) and verify that you can login as ‘John Doe’.

A black screen with white text

Description automatically generated

## Remote Windows server administration

We will now provide the tools to remotely manage our Windows Server from a Windows 11 client PC. We’ll do this in 3 different ways: using

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

# Via PowerShell

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 computer screen shot of a black screen

Description automatically generated

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

**5985, 5986**

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

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

**A screenshot of a computer

Description automatically generated**

**5985**

* + 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

* + 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 computer screen with text on it

Description automatically generated

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

A screen shot of a computer

Description automatically generated

* + Or actively verify open ports remotely on e.g. your laptop via a port scan with nmap (https://nmap.org/ , cfr module ‘Network and System Pentesting’).

A close up of a screen

Description automatically generated

Next, on the Windows 11 VM (and optionally on your Windows host OS):

* 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.

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer program

Description automatically generated

A computer screen shot of a computer program

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 king server to the TrustedHosts via the following in PowerShell (if in command prompt, omit the single quotes):

winrm set winrm/config/client '@{TrustedHosts="king,192.168.11.50"}'

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**

**A screen shot of a computer

Description automatically generated**

**That is cool but I am too lazy to do it, thank you very much though.**

* + - **Set-Item WSMan:\localhost\Client\TrustedHosts -Value "<servername>"**
    - **Clear-Item WSMan:\localhost\Client\TrustedHosts**
  + Via regedit.exe, you’ll find the same configurations at:
    - HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\WSMAN\
* You can now stop the WinRM service after changes in TrustedHosts are made, using Stop-Service WinRM , as the service only needs to run for incoming connections.

A screenshot of a computer program

Description automatically generated

Note: if you would like to also enable remote PowerShell on your Windows 11 VM, you can configure the WinRM service to auto start and to be allowed by the Windows Firewall via the winrm quickconfig command or the Enable-PSRemoting cmdlet.

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 king -Credential (Get-Credential)

Or Enter-PSSession -ComputerName 192.168.11.50 -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 computer screen shot of a black screen

Description automatically generated

* To leave the remote session and revert to the local PowerShell, type exit
* Optionally, if your laptop is running Windows 10/11: it’s convenient to also make remote PowerShell connections from your laptop to KING to manage it remotely (instead of via your Win11 VM). You could follow the same steps on your laptop host as you did on your Windows 11 VM (concerning the WinRM service and TrustedHosts).

# Via Server Manager

The standalone application ‘Server Manager’ is used a lot for managing a Windows Server. 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.

* Install the “RSAT – Server Manager” on your Windows 11 VM. This is available as a ‘Windows optional feature’ (via Settings – Apps – Optional Features).

Took insanely long, 3 fails via GUI, installed with PowerShell

* Start the Server Manager and right-click ‘All Servers’ and choose ‘Add Server’
* Click on the ‘DNS’ tab and search and add ‘king’
* It should succeed to add KING. However, retrieving status info will fail because you didn’t provide any credentials. Right click the server and choose ‘manage as’ and provide “king\Administrator” as username and his password.

A screenshot of a computer

Description automatically generated

It does not seem to work from my Windows 11 VM, and I do not know how to solve it for now. I tried to enable ICMPv4+6 rules, added the KING machine to trusted hosts, rebooted countless times, both machines see each other, but I could not make them work together. I had the same issue as in one of the labs from Datacenter Virtualization.

So I am doing this part from my laptop, as there is no other way to do it before the end of the week.

A screenshot of a computer

Description automatically generated

* Now, right click KING and try to open the ‘Computer Management’ for that server. You’ll get an error message concerning the Windows Firewall.
* Enable these rule(s). You can use the Enable-NetFirewallRule cmdlet at KING (locally or via a remote PowerShell session 😉).

A screenshot of a computer security system

Description automatically generated

**Enable-NetFirewallRule -DisplayGroup "Remote Event Log Management"**

* Now, you can manage KING via the Server Manager on your Windows 11 VM
* Optionally: If your laptop is running Windows 10/11 Professional/Enterprise/Education, you can also install RSAT Server Manager on your laptop host.
  + Note: if you install RSAT on your own laptop with Windows 10 1803 or earlier, you need to download RSAT separately at:

<https://www.microsoft.com/en-us/download/details.aspx?id=45520>

Choose the ‘WindowsTH-RSAT\_WS2016-*x64*.msu’ file, the 32-bit will not work on a 64-bit PC.

* + Note: RSAT can only be installed on Windows Professional/Enterprise/Education edition, not the Home edition.
  + Note: mind the TrustedHosts and WinRM service.
    - In case of WinRM errors in Server Manager, you need to make sure the server is added to your ‘TrustedHosts’ to connect to: cfr remarks above for connecting via PowerShell Remoting (with Enter-PSsession).

In my case, since I did it from my laptop host, here is how it looked like:

A computer screen shot of a black screen with white text

Description automatically generated

A screenshot of a computer

Description automatically generated

I also managed to get rid of the previous issue, however I had to set trusted hosts to “\*” and then it worked.

**winrm set winrm/config/client '@{TrustedHosts="\*"}'**

A screenshot of a computer

Description automatically generated

# Via Windows Admin Center

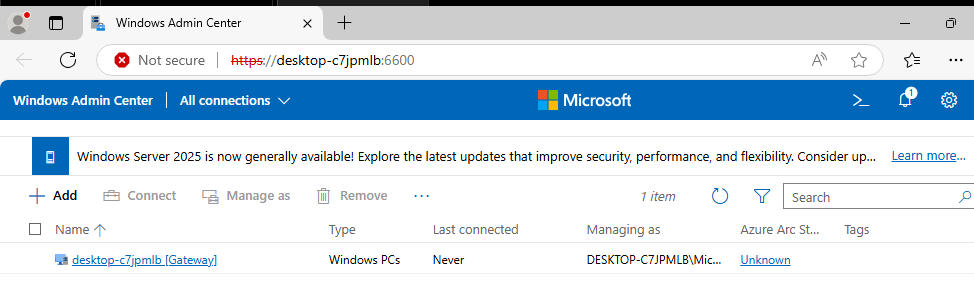
Microsoft allows administrators to manage their servers from a central location by providing them with a set of remote administration tools like Server Manager and RSAT/MMC plugins. These, however, typically lack integration, which means that the administrator needs to use a different tool, depending 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.

### Installing Windows Admin Center

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). Because we want to use the KING server as a DC (next lab), and we don’t have another dedicated server to use as a gateway, we’ll use our Windows 11 machine for a local client install of WAC.

* Copy the Windows Admin Center MSI file (which you’ve downloaded before) to your Windows 11 VM. A copy/paste between your host OS and your VM is possible because of the VMtools you’ve installed in your VM.
* Now start the install file (MSI) on your Windows 11 machine. Keep the install settings as they are proposed/recommended in the installer.
* Now restart your Windows 11 machine 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.



A screenshot of a computer

Description automatically generated

Looks kind of different from when I remember it, and I am not sure if it is using the right certificate or not, but whatever.

### Adding our King server to the managed servers

We’ll now add our “King” server to the list of devices, managed by 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 the IP address of the king server: 192.168.11.50

A screenshot of a computer

Description automatically generated

I added it with credentials king\Administrator

* Now the King server should appear in the list of managed devices. Select the checkbox to the left of the servername and click “manage as”. Use the “King\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!

Lets not forget that 1/3 of this functions is behind special firewall rules that do not allow me to do it from remote, but its fun anyway 😊