05/11/2024

TRAVAUX PRATIQUES HYPER-V MODE CORE

BTS SIO SISR 2ème année

Anne-Lou DELAGE-DAVIES
Kyllian AUCHER

Table des matières

Compte rendu du TP Hyper-V en mode Core

Introduction

Ce travail pratique s’inscrit dans le cadre de l’apprentissage des technologies de virtualisation avec **Hyper-V en mode Core**. Dans cet exercice, l'objectif était d'installer un serveur Hyper-V dépourvu d'interface graphique, d’y configurer deux machines virtuelles (l’une sous **Windows Server 2019** et l’autre sous **Ubuntu Server**) et de gérer ces machines à l'aide de commandes PowerShell. Cette approche offre une perspective différente et plus optimisée de la gestion des machines virtuelles, en se concentrant sur la ligne de commande et sur la performance, ce qui est particulièrement adapté aux environnements de production.

Matériel et logiciels utilisés

- Serveurs :

Un serveur Windows Server Core (versions compatibles avec Hyper-V).

Un serveur Windows Server pour le rôle Active Directory (AD-DS).

- Logiciels :

Hyper-V (rôle de virtualisation).

1. Objectifs

L'objectif principal de ce TP est de déployer un environnement de virtualisation en utilisant Hyper-V en mode Core. Ce projet consiste à configurer un serveur Hyper-V qui permettra de gérer efficacement deux machines virtuelles distinctes. La première machine virtuelle fonctionnera sous Windows Server 2019, offrant ainsi une plateforme pour tester les fonctionnalités avancées de ce système d'exploitation. La seconde machine virtuelle sera basée sur Ubuntu Server, permettant d'explorer les capacités de gestion et de configuration d'un environnement Linux. À travers ce TP, nous viserons à acquérir une compréhension pratique de la mise en place et de l'administration des machines virtuelles dans un environnement Hyper-V, tout en mettant en avant les compétences nécessaires pour effectuer des configurations adéquates et mener à bien des tests fonctionnels.

2. Installation d'Hyper-V en mode Core

2.1 Installation d'Hyper-V

L'installation a débuté par la mise en place du rôle Hyper-V sur un serveur **Windows Server Core**. Pour ce faire, j'ai utilisé PowerShell avec la commande suivante :

```
Install-WindowsFeature -Name Hyper-V -IncludeManagementTools -Restart
```

Question 1

*L'installation en mode Core présente des avantages non négligeables. D'une part, la **consommation de ressources** est nettement réduite, car le serveur n'a pas à gérer une interface graphique, libérant ainsi des ressources pour les machines virtuelles elles-mêmes. D'autre part, l'aspect **sécuritaire** est renforcé : moins de services et de composants installés signifient une surface d'attaque plus faible et un risque réduit de vulnérabilités.*

3. Paramétrage Hyper-V mode Core

Nous avons alors changé le nom NetBios du serveur comme ceci :

```
Rename-Computer -NewName "hyperv-core" -Restart
```

3.1 Configuration réseau

Après l'installation, le serveur a redémarré pour appliquer les modifications. Ensuite, j'ai procédé à la configuration réseau de l'hyperviseur en définissant une adresse IP statique (**192.168.10.2**), la passerelle (**192.168.10.1**) et le DNS. Cela a permis d'intégrer le serveur au domaine **LABO1.local** pour que l'Active Directory puisse reconnaître et gérer ce serveur Hyper-V.

Voici les commandes :

```
New-NetIPAddress -InterfaceAlias "Ethernet" -IPAddress "192.168.10.2" -PrefixLength 24 -DefaultGateway "192.168.10.1"
```

```
Set-DnsClientServerAddress -InterfaceAlias "Ethernet" -ServerAddresses "192.168.10.1"
```

```
Add-Computer -DomainName "LABO1.local" -Credential "LABO1\Administrateur" -Restart
```

On effectue ensuite les mises à jour Windows avec :

```
Install-WindowsUpdate
```

Et on vérifie notre configuration :

```
Get-NetIPAddress
```

3.2 Joindre le domaine

Pour que le serveur soit intégré au domaine **LABO1.local**, j'ai utilisé la commande suivante :

```
Add-Computer -DomainName "LABO1.local" -Credential LABO1\AdminUser
```

Après avoir exécuté cette commande, j'ai redémarré le serveur avec :

```
Restart-Computer
```

4. Installation machine virtuelle Windows serveur

Installation machine virtuelle Windows serveur

Avec Hyper-V installé, j'ai créé la première machine virtuelle, nommée **VM-WIN-SRV-01**, avec les spécifications suivantes : **2 Go de RAM**, **100 Go de disque dur**, et **2 processeurs**. La commande utilisée pour cela a été :

```
New-VM -Name "VM-WIN-SRV-01" -MemoryStartupBytes 2GB -Generation 2 -NewVHDPath "C:\VMs\VM-WIN-SRV-01.vhdx" -NewVHDSizeBytes 100GB
Set-VMProcessor -VMName "VM-WIN-SRV-01" -Count 2
Set-VMDvdDrive -VMName "VM-WIN-SRV-01" -Path "C:\ISOs\Windows2019.iso"
```

Pour démarrer, arrêter ou redémarrer notre VM, nous utilisons les commandes :

```
Start-VM -Name "VM-WIN-SRV-01"
Stop-VM -Name "VM-WIN-SRV-01"
Restart-VM -Name "VM-WIN-SRV-01"
```

Après avoir créé la machine virtuelle, j'ai configuré un **commutateur virtuel interne** nommé **RES-INT-1** pour permettre à cette machine de communiquer avec le réseau interne de l'hyperviseur. La création et l'assignation du commutateur à la VM a été faite avec la commande suivante :

```
New-VMSwitch -Name "RES-INT-1" -SwitchType Internal
Connect-VMNetworkAdapter -VMName "VM-WIN-SRV-01" -SwitchName "RES-INT-1"
```

Par la suite, j'ai monté l'image ISO de Windows Server 2019 et j'ai démarré la machine pour procéder à son installation. Une fois l'installation terminée, j'ai pu vérifier que la machine virtuelle fonctionnait correctement avec des ressources adéquates.

4.2 Création d'un instantané et gestion de la machine virtuelle

Pour sécuriser l'état de la machine virtuelle avant d'y apporter des modifications, j'ai créé un **instantané**, aussi appelé **snapshot**, afin de pouvoir revenir à cette configuration initiale en cas de besoin :

```
Checkpoint-VM -Name "VM-WIN-SRV-01" -SnapshotName "avantmaj"
```

Pour afficher tous les instantanés de la machine virtuelle, j'ai utilisé :

```
Get-VMSnapshot -VMName "VM-WIN-SRV-01"
```

Cet instantané a été d'une grande utilité lors des tests de manipulation, car j'ai pu restaurer l'état de la machine plusieurs fois sans perdre les configurations importantes. Par exemple, pour restaurer la machine à l'instantané, j'ai utilisé :

```
Restore-VMSnapshot -VMName "VM-WIN-SRV-01" -Name "avantmaj"
```

Pour supprimer l'instantané créé précédemment, j'ai utilisé :

```
Remove-VMSnapshot -VMName "VM-WIN-SRV-01" -Name "avantmaj"
```

4.3 Duplication de la machine virtuelle : création de VM-WIN-SRV-02

L'une des étapes du TP impliquait la **duplication de VM-WIN-SRV-01** pour créer une seconde instance, **VM-WIN-SRV-02**. Cela a posé un défi : Hyper-V identifie les machines virtuelles par un **GUID unique** et refuse d'importer une VM si une autre machine avec le même GUID est présente. Pour contourner ce problème, j'ai d'abord exporté **VM-WIN-SRV-01** :

```
Export-VM -Name "VM-WIN-SRV-01" -Path "C:\Hyper-V\ExportedVMs"
```

Ensuite, j'ai importé la VM via l'interface graphique d'Hyper-V depuis une autre machine, ce qui m'a permis de créer un nouveau GUID et d'importer la machine avec un nouveau nom. Cette étape a été cruciale pour comprendre l'importance des identifiants de VM dans Hyper-V et a servi de bonne introduction à la gestion des imports et exports de machines virtuelles.

5. Installation machine virtuelle Linux serveur

Avec Hyper-V installé, j'ai créé la première machine virtuelle, nommée **VM-LIN-SRV-01**, avec les spécifications suivantes : **2 Go de RAM**, **40Go de disque dur**, et 1 **processeur**. La commande utilisée pour cela a été :

```
New-VM -Name "VM-LIN-SRV-01" -MemoryStartupBytes 2GB -NewVHDPath "C:\Hyper-V\VM-LIN-SRV-01.vhdx" -NewVHDSizeBytes 40GB -SwitchName "RES-INT-1" -Generation 1
```

Montage de l'image ISO et démarrage de l'installation

Après avoir créé la machine virtuelle, j'ai monté l'image ISO d'Ubuntu Server pour lancer l'installation.

Voici la commande utilisée pour cela :

```
Set-VMDvdDrive -VMName "VM-LIN-SRV-01" -Path "C:\Path\To\Your\ISO\ubuntu-server.iso"
```

Ensuite, j’ai démarré la machine virtuelle pour procéder à l’installation d’Ubuntu Server :

```
Start-VM -Name "VM-LIN-SRV-01"
```

6. Manipulations

1. Affichage des machines virtuelles

Pour afficher toutes les machines virtuelles sur l'hyperviseur, j'ai utilisé la commande suivante :

```
Get-VM
```

2. Détails de la machine virtuelle VM-WIN-SRV-01

Affichage de VM-WIN-SRV-01

Pour afficher les détails de la machine virtuelle VM-WIN-SRV-01, j'ai exécuté :

```
Get-VM -Name "VM-WIN-SRV-01"
```

Nombre de processeurs alloués

Le nombre de processeurs alloués à cette machine virtuelle a été vérifié avec :

```
Get-VMProcessor -VMName "VM-WIN-SRV-01"
```

Mémoire vive allouée

La mémoire vive allouée à VM-WIN-SRV-01 a été affichée grâce à :

```
Get-VMMemory -VMName "VM-WIN-SRV-01"
```

3. Gestion des instantanés

Création d'un instantané

J'ai créé un instantané appelé "avantmaj" avec la commande :

```
Checkpoint-VM -Name "VM-WIN-SRV-01" -SnapshotName "avantmaj"
```

Affichage des instantanés

Les instantanés de VM-WIN-SRV-01 ont été affichés avec :

```
Get-VMSnapshot -VMName "VM-WIN-SRV-01"
```

Restauration de l'instantané

Pour restaurer l'instantané "avantmaj", j'ai utilisé :

```
Restore-VMSnapshot -VMName "VM-WIN-SRV-01" -Name "avantmaj"
```

Suppression de l'instantané

L'instantané "avantmaj" a été supprimé avec :

```
Remove-VMSnapshot -VMName "VM-WIN-SRV-01" -Name "avantmaj"
```

4. Gestion des états des machines virtuelles

Suspension de VM-WIN-SRV-01

J'ai suspendu la machine virtuelle VM-WIN-SRV-01 avec la commande :

```
Suspend-VM -Name "VM-WIN-SRV-01"
```

Arrêt de VM-LIN-SRV-01

La machine virtuelle VM-LIN-SRV-01 a été arrêtée grâce à :

```
Stop-VM -Name "VM-LIN-SRV-01"
```

Débranchement de VM-LIN-SRV-01

Pour débrancher VM-LIN-SRV-01, j'ai utilisé :

```
Stop-VM -Name "VM-LIN-SRV-01" -TurnOff
```

Démarrage des machines virtuelles

Les machines virtuelles VM-WIN-SRV-02 et VM-LIN-SRV-02 ont été démarrées avec :

```
Start-VM -Name "VM-WIN-SRV-02"
Start-VM -Name "VM-LIN-SRV-02"
```

5. États des machines virtuelles

Affichage des machines virtuelles en fonctionnement

Les machines virtuelles en fonctionnement ont été listées par :

```
Get-VM | Where-Object {$_.State -eq 'Running'}
```

Affichage des machines virtuelles suspendues

Les machines suspendues ont été affichées avec :

```
Get-VM | Where-Object {$_.State -eq 'Paused'}
```

Affichage des machines virtuelles arrêtées

Les machines arrêtées ont été listées avec :

```
Get-VM | Where-Object {$_.State -eq 'Off'}
```

6. Modification des ressources de VM-LIN-SRV-02

Assignation de ressources

J'ai assigné 2 processeurs et 3 Go de mémoire vive à la machine virtuelle VM-LIN-SRV-02 avec les commandes suivantes :

```
Set-VMProcessor -VMName "VM-LIN-SRV-02" -Count 2
Set-VMMemory -VMName "VM-LIN-SRV-02" -StartupBytes 3GB
```

Suppression de la machine virtuelle

La machine virtuelle VM-LIN-SRV-02 a été supprimée avec :

```
Remove-VM -Name "VM-LIN-SRV-02"
```

7. Gestion des interfaces réseau

Affichage des interfaces réseau

Pour afficher les interfaces réseau de l’hyperviseur, j'ai utilisé :

```
Get-NetAdapter
```

Création des interfaces réseau virtuelles

Les interfaces réseau virtuelles "sw-test1" et "sw-test2" en mode externe ont été créées avec :

```
New-VMSwitch -Name "sw-test1" -NetAdapterName "Ethernet" -AllowManagementOS $true
New-VMSwitch -Name "sw-test2" -NetAdapterName "Ethernet" -AllowManagementOS $true
```

Affichage des interfaces réseau virtuelles

J'ai affiché les interfaces réseau virtuelles avec :

```
Get-VMSwitch
```

Ajout des interfaces réseau

Les interfaces réseau virtuelles ont été ajoutées aux machines virtuelles :

```
Connect-VMNetworkAdapter -VMName "VM-WIN-SRV-01" -SwitchName "sw-test1"
Connect-VMNetworkAdapter -VMName "VM-LIN-SRV-01" -SwitchName "sw-test2"
```

Affichage des interfaces réseau de VM-WIN-SRV-01

Pour afficher les interfaces réseau virtuelles de VM-WIN-SRV-01, j'ai exécuté :

```
Get-VMNetworkAdapter -VMName "VM-WIN-SRV-01"
```

Suppression des interfaces réseau de VM-LIN-SRV-01

Les interfaces réseau virtuelles de VM-LIN-SRV-01 ont été supprimées avec :

```
Disconnect-VMNetworkAdapter -VMName "VM-LIN-SRV-01"
```

Suppression de l'interface réseau virtuelle sw-test1

Enfin, l'interface réseau virtuelle "sw-test1" a été supprimée de l'Hyper-V avec :

```
Remove-VMSwitch -Name "sw-test1"
```

Problèmes et solutions

Un des problèmes majeurs rencontrés a été lié aux **droits d'accès**. En tentant de me connecter au serveur Hyper-V Core depuis le serveur AD, une erreur de permission est survenue. Cela a été résolu en ajoutant le compte utilisateur aux **Administrateurs locaux** sur le serveur Hyper-V et en s'assurant que les deux serveurs pouvaient communiquer correctement en désactivant temporairement le pare-feu.

Commandes utilisées pour la résolution :

Ajouter un utilisateur au groupe Administrateurs :

```
Add-LocalGroupMember -Group "Administrators" -Member "Domaine\NomUtilisateur"
```

Désactiver temporairement le pare-feu :

```
Set-NetFirewallProfile -Profile Domain,Public,Private -Enabled False
```

Un autre problème a été la **non-résolution de nom**. Les serveurs ne se pingaient pas initialement. J'ai donc dû vérifier les configurations réseau, m'assurer que les interfaces correctes étaient en place et que les pare-feux autorisaient les connexions ICMP. Après ces ajustements, la communication entre les serveurs a pu se faire correctement.

Commande pour tester la connectivité :

```
Test-Connection -ComputerName "Adresse_IP_HyperV_Core" -Count 4
```

7. Commandes bonus

1. Gestion des instantanés

1.1 Création d'un instantané pour toutes les VM

Pour créer un instantané pour toutes les machines virtuelles en une seule commande, j'ai exécuté :

```
Get-VM | ForEach-Object { Checkpoint-VM -VMName $_.Name -SnapshotName "snapshot-all" }
```

Cette commande crée un instantané nommé "snapshot-all" pour chaque VM, facilitant ainsi la gestion des sauvegardes.

1.2 Suppression des instantanés de toutes les VM

Pour supprimer tous les instantanés existants, j'ai utilisé la commande suivante :

```
Get-VM | ForEach-Object { Get-VMSnapshot -VMName $_.Name | Remove-VMSnapshot }
```

Cette opération nettoie les instantanés, libérant de l'espace et évitant l'encombrement des ressources.

2. Gestion des Machines Virtuelles

2.1 Arrêt de toutes les VM

Pour arrêter toutes les machines virtuelles simultanément, la commande suivante a été utilisée :

```
Get-VM | ForEach-Object { Stop-VM -Name $_.Name -Force }
```

Cela permet une gestion rapide et efficace des VMs en cas de besoin.

2.2 Arrêt des VM commençant par "Win"

Pour cibler uniquement les machines virtuelles dont le nom commence par "Win", j'ai exécuté :

```
Get-VM | Where-Object { $_.Name -like "VM-Win*" } | ForEach-Object { Stop-VM -Name $_.Name -Force }
```

Cette commande assure que seules les VMs Windows sont arrêtées, tout en laissant les autres en fonctionnement.

2.3 Démarrage de toutes les VM

Pour redémarrer toutes les machines virtuelles, j'ai utilisé :

```
Get-VM | ForEach-Object { Start-VM -Name $_.Name }
```

Ceci permet de remettre en service toutes les VMs en un seul appel.

2.4 Démarrage des VMs arrêtées

Pour démarrer uniquement les VMs qui sont actuellement arrêtées, j'ai utilisé :

```
Get-VM | Where-Object {$_.State -eq 'Off'} | ForEach-Object { Start-VM -Name $_.Name }
```

Cela permet de relancer rapidement les machines qui n'étaient pas en fonctionnement.

3. Planification des tâches

3.1 Tâche planifiée pour arrêter la VM Windows

Pour automatiser l'arrêt de la VM **VM-WIN-SRV-01** tous les jours à 23h00, j'ai créé une tâche planifiée :

```
$action = New-ScheduledTaskAction -Execute "powershell.exe" -Argument "Stop-VM -Name 'VM-WIN-SRV-01' -Force"
$trigger = New-ScheduledTaskTrigger -Daily -At "23:00"
Register-ScheduledTask -Action $action -Trigger $trigger -TaskName "Stop-Windows-VM" -Description "Arrête la VM Windows tous les jours à 23h00"
```

Cela garantit que la machine virtuelle est arrêtée à une heure prédéfinie, facilitant la gestion des ressources.

3.2 Tâche planifiée pour importer la VM Linux

Pour importer automatiquement la VM **VM-LIN-SRV-01** tous les dimanches, j'ai configuré la tâche suivante :

```
$action = New-ScheduledTaskAction -Execute "powershell.exe" -Argument "Import-VM -Path
```

```
'C:\VMExport\VM-LIN-SRV-01'"
$trigger = New-ScheduledTaskTrigger -Weekly -DaysOfWeek Sunday -At "00:00"
Register-ScheduledTask -Action $action -Trigger $trigger -TaskName "Import-Linux-VM" -Description "Importe la VM Linux tous les dimanches"
```

Cela automatise le processus d'importation, rendant la gestion des VMs plus fluide.

4. Gestion des services et processus

4.1 Voir les services en cours d'exécution

Pour consulter les services actifs, j'ai utilisé :

```
Get-Service | Where-Object { $_.Status -eq 'Running' }
```

Cela permet d'obtenir une vue d'ensemble des services en fonctionnement sur le système.

4.2 Voir les services arrêtés

Pour identifier les services qui ne fonctionnent pas, la commande suivante a été exécutée :

```
Get-Service | Where-Object { $_.Status -eq 'Stopped' }
```

4.3 Arrêter et lancer le service de mise à jour

Pour gérer le service Windows Update, j'ai utilisé :

```
Stop-Service -Name "wuauserv" -Force
Start-Service -Name "wuauserv"
```

Ces commandes permettent de contrôler la mise à jour du système selon les besoins.

4.4 Gestion des processus

Pour afficher tous les processus en cours d'exécution :

```
Get-Process
```

Pour cibler le processus Firefox :

```
Get-Process -Name "firefox"
```

Et pour arrêter le processus de l'explorateur :

```
Stop-Process -Name "explorer" -Force
```

Pour le relancer :

```
Start-Process -FilePath "explorer.exe"
```

5. Informations Système

5.1 Informations sur le Disque Physique

Pour obtenir des informations sur les disques physiques :

```
Get-PhysicalDisk
```

5.2 Informations sur le Matériel

Pour consulter les détails matériels :

```
Get-WmiObject -Class Win32_ComputerSystem
```

5.3 Informations sur l'OS et le Processeur

Pour obtenir des informations sur le système d'exploitation :

```
Get-WmiObject -Class Win32_OperatingSystem
```

Et pour le processeur :

```
Get-WmiObject -Class Win32_Processor
```

5.4 Ressources Système

Pour obtenir des détails spécifiques :

Nombre de cœurs du processeur :

```
(Get-WmiObject -Class Win32_Processor).NumberOfCores
```

Nombre de processeurs physiques :

```
(Get-WmiObject -Class Win32_ComputerSystem).NumberOfProcessors
```

Mémoire totale :

```
(Get-WmiObject -Class Win32_ComputerSystem).TotalPhysicalMemory / 1GB
```

Mémoire disponible :

```
(Get-WmiObject -Class Win32_OperatingSystem).FreePhysicalMemory / 1024
```

Nom NetBIOS :

```
(Get-WmiObject -Class Win32_ComputerSystem).Name
```

Modèle et Fabricant :

```
(Get-WmiObject -Class Win32_ComputerSystem).Model
(Get-WmiObject -Class Win32_ComputerSystem).Manufacturer
```

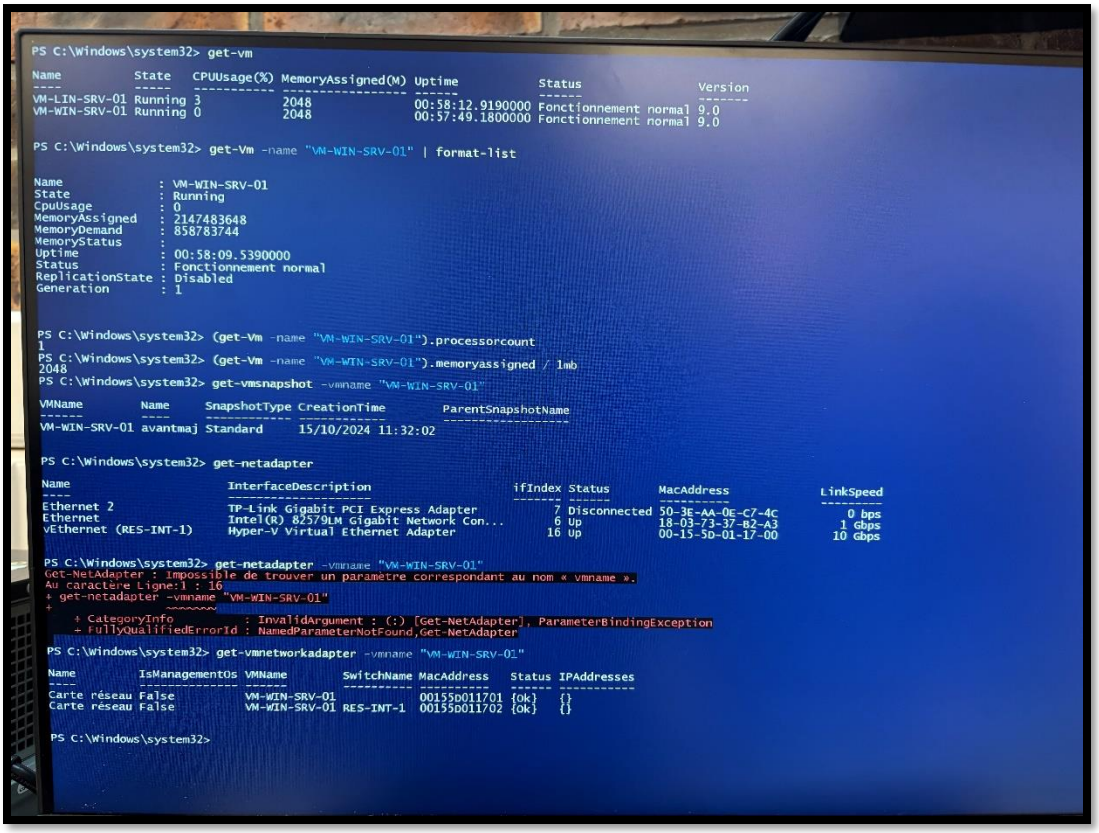
Cartes réseau :

```
Get-CimInstance -ClassName Win32_NetworkAdapterConfiguration | Where-Object { $_.IPEnabled -eq $true }
```

Conclusion

Ce TP m’a permis de découvrir la **gestion avancée de la virtualisation en mode Core** et de mieux comprendre les configurations réseau et la gestion des ressources dans Hyper-V. Les erreurs rencontrées et résolues m'ont également apporté une vision pratique des défis d’un environnement virtualisé. Ce TP est un excellent point de départ pour ceux qui souhaitent approfondir leur compréhension de la virtualisation et des environnements sans interface graphique.

Annexes



Nos VM sur le serveur mode Core