**Introduction aux cmdlets de PowerShell pour la gestion TCP/IP**

**Objectifs**

* Utiliser les principaux cmdlets qui permettent de configurer plusieurs paramètres TCP/IP.

**Documentation**

* L'aide de PowerShell

**Travail**

* L'étudiant est invité à prendre des notes à la suite de son expérience avec ces cmdlets.
* L'étudiant doit consulter l'aide de PowerShell pour documenter et améliorer sa connaissance des cmdlets qui sont dans ce laboratoire.

**L’adresse IP 10.57.x.y correspond à l’adresse IP de votre serveur réel.**

**À moins d’avis contraire, vous pouvez exécuter les commandes PowerShell sur le SERVEUR1 ou sur le SERVEUR2.**

**Introduction au cmdlet Get-Command**

**Pour afficher la liste des cmdlets d'un module**

(Get-Command -Module **NetConnection**).Name

**Pour afficher le nom du module qui contient un cmdlet donné**

Get-Command -Name **Test-Connection** | Select-Object Module

**Pour afficher le nom des cmdlets qui contiennent IPADDRESS avec le nom du module des cmdlets**

Get-Command -Name **\*ipaddress\*** | Select-Object Name,Module

**Exercice 1 - Étude des deux cmdlets du module NetConnection**

Le module NetConnection contient seulement deux cmdlets.

* Get-NetConnectionProfile
* Set-NetConnectionProfile

Exécuter les commandes suivantes:

**Get-NetConnectionProfile**

**Get-NetConnectionProfile -InterfaceAlias "Ethernet0"**

Cette commande modifie le type de réseau à "Privé" pour une carte réseau donnée

**Set-NetConnectionProfile -InterfaceAlias "Ethernet0" -NetworkCategory Private**

Cette commande modifie le type de réseau à "Public" pour une carte réseau donnée

**Set-NetConnectionProfile -InterfaceAlias "Ethernet0" -NetworkCategory Public**

**Exercice 2 - Étude du cmdlet Test-Connection**

Le module Microsoft.PowerShell.Management contient le cmdlet Test-Connection

* Test-Connection ce cmdlet est similaire à la commande **ping**

**Vous devez exécuter les commandes PowerShell sur le SERVEUR1.**

**Le temps d'exécution de Test-Connection est lent si la destination n'est pas connectée à un domaine.**

**Test-Connection www.google.com # le temps d'exécution est rapide**

**Test-connection uranus.reseau.cvm # le temps d'exécution est rapide**

**Test-connection 10.19.3.106 # le temps d'exécution est rapide**

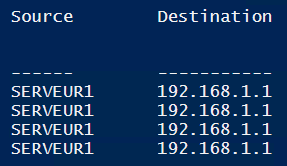
**Test-Connection 192.168.1.1 # le temps d'exécution est lent**

Chaque commande envoie les pings de l’ordinateur local à un ordinateur distant.

**Test-Connection -Source 192.168.1.10 -ComputerName 192.168.1.1**

**Test-Connection -ComputerName 192.168.1.1**

**Test-Connection 192.168.1.1**

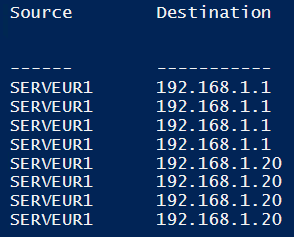


Chaque commande envoie les pings de l’ordinateur local à plusieurs ordinateurs distants.

**Test-Connection -Source 192.168.1.10 -ComputerName 192.168.1.1,192.168.1.20**

**Test-Connection -ComputerName 192.168.1.1,192.168.1.20**

**Test-Connection 192.168.1.1,192.168.1.20**



Test-Connection n'a pas de paramètre équivalent au paramètre -t de la commande PING.

**while ($true) { Test-Connection -ComputerName www.google.com -Count 1 }**

**Exercice 3 - Étude de plusieurs cmdlets du module DnsClient**

Voici plusieurs cmdlets utiles du module DnsClient

* Get-DnsClientServerAddress
* Resolve-DNSName

Resolve-DNSName retourne l'adresse IPv4 (type A) et l'adresse IPv6 (type AAAA) d'un nom de domaine.

Par défaut, la commande Resolve-DNSName utilise le serveur DNS de votre carte réseau

La commande affiche l'adresse IPv4 et l'adresse IPv6 de www.facebook.com.

La commande utilise le serveur DNS préféré de votre carte réseau.

**Resolve-DNSName -Name www.facebook.com**

La commande affiche l'adresse IPv4 et l'adresse IPv6 de www.facebook.com.

La commande utilise le serveur DNS 8.8.8.8.

**Resolve-DNSName -Name www.facebook.com -Server 8.8.8.8**

La commande seulement l'adresse IPv4 de www.facebook.com

**Resolve-DNSName -Name www.facebook.com -Type A**

La commande seulement l'adresse IPv6 de www.facebook.com

**Resolve-DNSName -Name www.facebook.com -Type AAAA**

**Exercice 4a - Étude de plusieurs cmdlets du module NetTCPIP**

Voici plusieurs cmdlets utiles du module NetTCPIP

* Get-NetIPConfiguration
* Get-NetNeighbor
* Test-NetConnection

Cette commande affiche la configuration de toutes les cartes réseau

**Get-NetIPConfiguration**

Cette commande affiche la configuration d'une carte réseau particulière

**Get-NetIPConfiguration -InterfaceAlias "Ethernet0"**

Cette commande affiche seulement les adresses IPv4 d'un ordinateur

**((Get-NetIPConfiguration).IPv4Address).IPAddress | Sort-Object**

Cette commande affiche le contenu de la table ARP mais seulement pour les adresses IPv4

**Get-NetNeighbor -AddressFamily IPv4**

Cette commande vérifie si le PING a fonctionné sur un ordinateur et PingSucceeded retourne TRUE ou FALSE

**(Test-NetConnection -ComputerName 10.57.x.y).PingSucceeded**

Cette commande vérifie si le "Bureau à distance" est accessible sur un ordinateur

**Test-NetConnection -ComputerName 10.57.x.y -CommonTCPPort RDP**

Cette commande vérifie si le port 80 est accessible sur un ordinateur

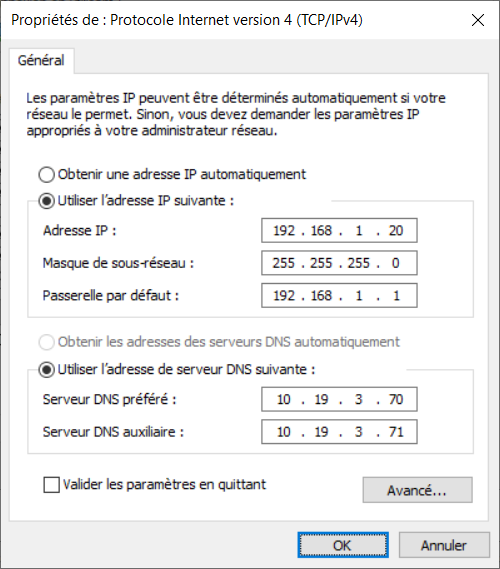
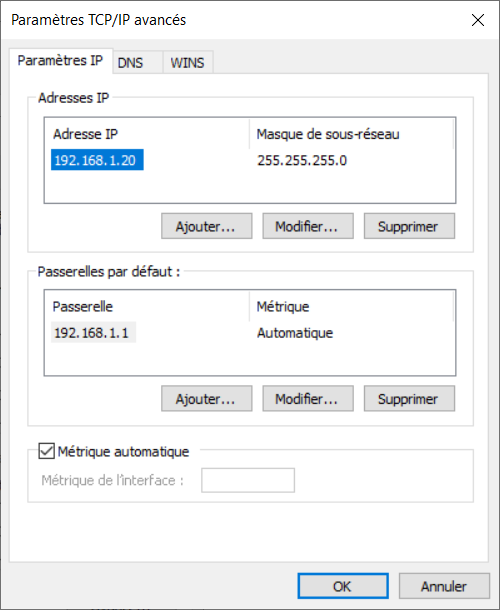
**Test-NetConnection -ComputerName 10.57.x.y -Port 80**

**Exercice 4b - Étude de plusieurs cmdlets du module NetTCPIP**

Voici plusieurs cmdlets utiles du module NetTCPIP

* Get-NetIPAddress
* Get-NetIPInterface
* New-NetIPAddress
* Remove-NetIPAddress
* Set-NetIPAddress
* Set-NetIPInterface

**Exemples pour récupérer la configuration IP d’une carte réseau**

La configuration initiale de la carte réseau "**Ethernet0**". La carte réseau "**Ethernet0**" a une seule adresse IP.

**Vous devez exécuter les commandes PowerShell sur le SERVEUR2.**

# Cette commande permet d’obtenir l’adresse IP et le masque de sous-réseau de la carte réseau "**Ethernet0**".

**Get-NetIPAddress -InterfaceAlias "Ethernet0" -AddressFamily IPv4**

# Cette commande affiche les adresses des serveurs DNS de la carte réseau "**Ethernet0**".

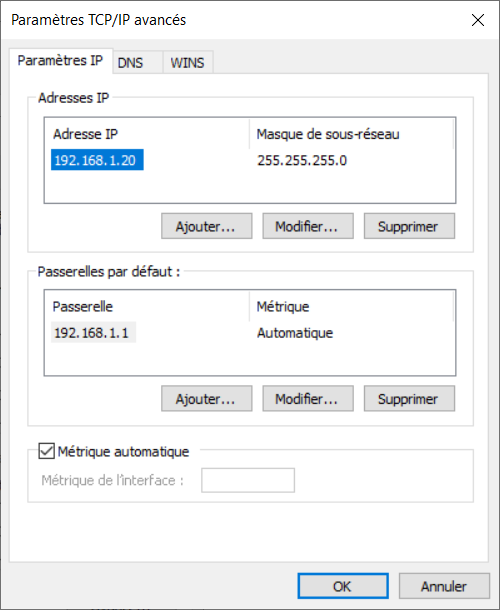
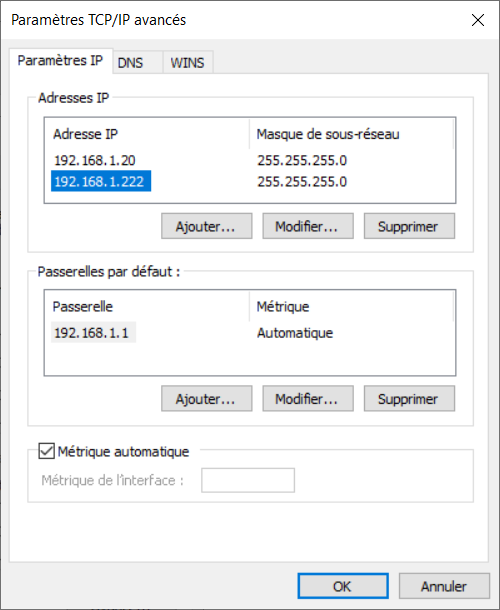
**Get-DnsClientServerAddress -InterfaceAlias "Ethernet0" -AddressFamily IPv4**

# Cette commande affiche la passerelle par défaut de la carte réseau "**Ethernet0**".

**$resultat = (Get-NetIPConfiguration -InterfaceAlias** "**Ethernet0**"**).Ipv4DefaultGateway**

**$passerelle = $resultat.NextHop**

**Exemples pour ajouter une adresse IP sur une carte réseau**

La configuration initiale de la carte réseau "**Ethernet0**". La configuration finale de la carte réseau "**Ethernet0**".

La carte réseau "**Ethernet0**" a une seule adresse IP. La carte réseau "**Ethernet0**" a deux adresses IP.

**Vous devez exécuter les commandes PowerShell sur le SERVEUR2.**

# Cette commande ajoute l'adresse IP 192.168.1.222/24 à la carte réseau "**Ethernet0**".

**New-NetIPAddress -InterfaceAlias "Ethernet0" `**

**-AddressFamily "IPv4" `**

**-IPAddress 192.168.1.222 `**

**-PrefixLength 24**

# Cette commande affiche les adresses IP de la carte réseau "**Ethernet0**".

# La commande affiche les adresses IP 192.168.1.20 et 192.168.1.222

**(Get-NetIPAddress -InterfaceAlias "Ethernet0" -AddressFamily IPv4).IPAddress**

# Cette commande supprime l'adresse IP 192.168.1.222 de la carte réseau "**Ethernet0**".

# Le paramètre **-Confirm:$false** force la suppression de l'adresse IP sans demander de confirmation

**Remove-NetIPAddress -InterfaceAlias "Ethernet0" `**

**-AddressFamily "IPv4" `**

**-IPAddress 192.168.1.222 `**

**-Confirm:$false**

# Cette commande affiche les adresses IP de la carte réseau "**Ethernet0**".

# La commande affiche seulement l’adresse IP 192.168.1.20

**(Get-NetIPAddress -InterfaceAlias "Ethernet0" -AddressFamily IPv4).IPAddress**

**Exercice 5 - Étude de plusieurs cmdlets du module NetAdapter**

Voici plusieurs cmdlets utiles du module NetAdapter

* Disable-NetAdapter
* Enable-NetAdapter
* Get-NetAdapter
* Get-NetAdapterStatistics
* Rename-NetAdapter
* Restart-NetAdapter
* Set-NetAdapter

**Get-NetAdapter -Name \***

**Get-NetAdapter -Name \* -Physical**

**Get-NetAdapter -Name \* -Physical | Select-Object Name,MacAddress**

**Exercice 6 - Comment trouver l'adresse IPv4 de votre fournisseur internet**

**Invoke-WebRequest est un cmdlet du module Microsoft.PowerShell.Utility**

**# Premier exemple**

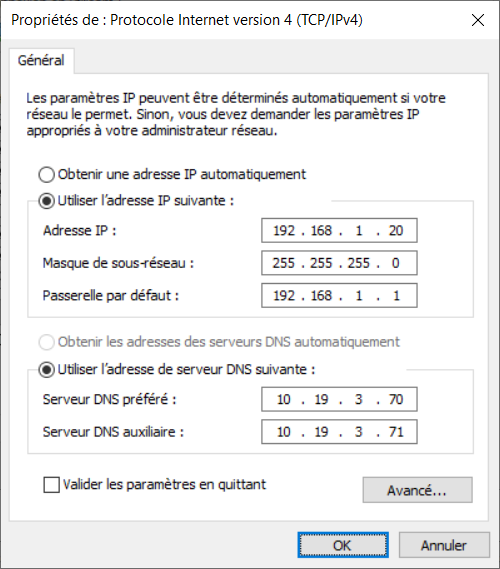
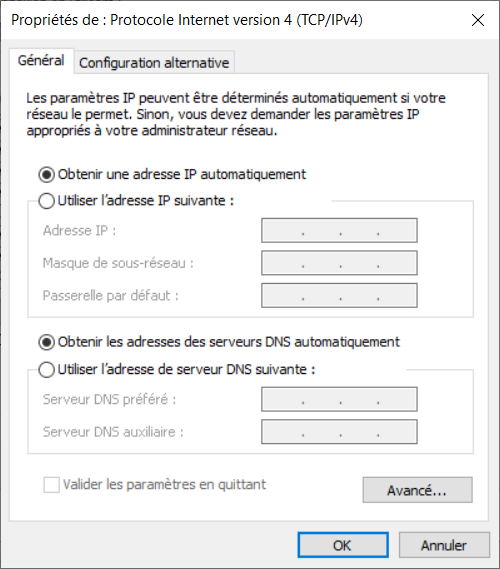
**(Invoke-WebRequest -Uri https://ifconfig.me/ip).Content**

**# Deuxième exemple**

**(Invoke-WebRequest -Uri https://myexternalip.com/raw).Content**

**ANNEXE 1**

**Rétablir une configuration par DHCP sur une carte réseau**

La configuration initiale de la carte réseau "**Ethernet0**". La configuration finale après l’exécution des commandes.

**Vous devez exécuter les commandes PowerShell sur le SERVEUR2.**

**$NIC =** **"Ethernet0"**

**# Active l’option "Obtenir une adresse IP automatiquement" sur la carte réseau**

**Set-NetIPInterface -InterfaceAlias $NIC -Dhcp Enabled**

**# Active l’option "Obtenir les adresses des serveurs automatiquement" sur la carte réseau**

**Set-DnsClientServerAddress -InterfaceAlias $NIC -ResetServerAddresses**

**# La variable $passerelle va contenir la passerelle par défaut de la carte réseau**

**$resultat = (Get-NetIPConfiguration -InterfaceAlias $NIC).Ipv4DefaultGateway**

**$passerelle = $resultat.NextHop**

**If ($passerelle -ne $null)**

**{**

**$info = "Suppression de la passerelle par défaut $passerelle"**

**Write-Host $info -ForegroundColor Cyan**

**# Supprime la passerelle par défaut de la carte réseau**

**Remove-NetRoute -InterfaceAlias $NIC -Confirm:$false**

**}**

**else**

**{**

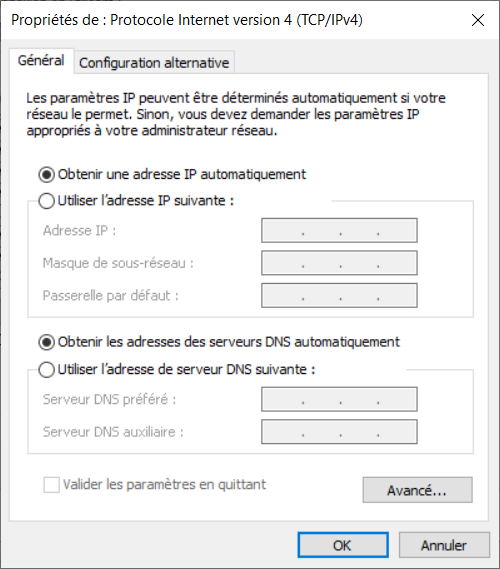
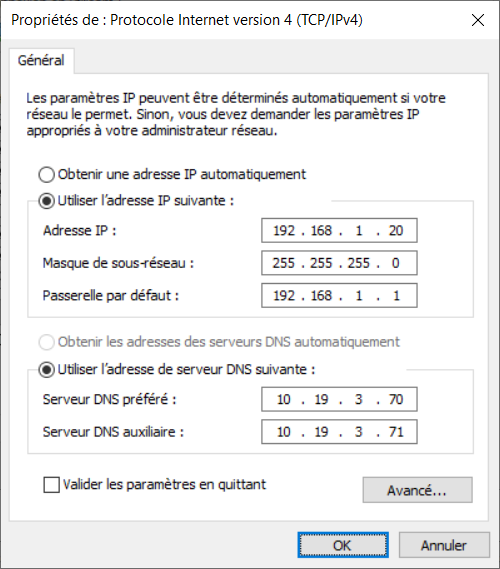
**$info = "Aucune passerelle"**

**Write-Host $info -ForegroundColor Yellow**

**}**

**ANNEXE 2**

**Rétablir une configuration statique sur une carte réseau**

La configuration initiale de la carte réseau "**Ethernet0**". La configuration finale après l’exécution des commandes.

**Vous devez exécuter les commandes PowerShell sur le SERVEUR2.**

**$NIC = "Ethernet0"**

**$IP = "192.168.1.20"**

**$MaskBits = 24**

**$Gateway = "192.168.1.1"**

**$Dns = "10.19.3.70,10.19.3.71"**

**$IPType = "IPv4"**

**# Configuration de l’adresse IP, du masque de sous-réseau et de la passerelle par défaut**

**New-NetIPAddress -InterfaceAlias $NIC `**

**-AddressFamily $IPType `**

**-IPAddress $IP `**

**-PrefixLength $MaskBits `**

**-DefaultGateway $Gateway**

**# Configuration de l’adresse de serveur DNS**

**Set-DnsClientServerAddress -InterfaceAlias $NIC -ServerAddresses $DNS**

**ANNEXE 3**

**Activation du PSRemoting pour exécuter du code à distance sur un autre serveur**

**Si des ordinateurs ne sont pas membres d'un domaine "Active Directory" mais d'un "Groupe de travail", c'est plus difficile d'avoir accès à un autre ordinateur par programmation PowerShell.**

**Vous devez exécuter les prochaines commandes PowerShell sur le SERVEUR1 et sur le SERVEUR2.**

**Enable-PSRemoting -SkipNetworkProfileCheck -Force**

# Cette commande démarre le service WinRM

# Active la fonctionnalité "**Gestion à distance de Windows**" dans le "Pare-feu Windows".

**Set-Item WSMan:\localhost\Client\TrustedHosts -Value \* -Force**

# Cette commande ajoute des ordinateurs auxquels on a confiance.

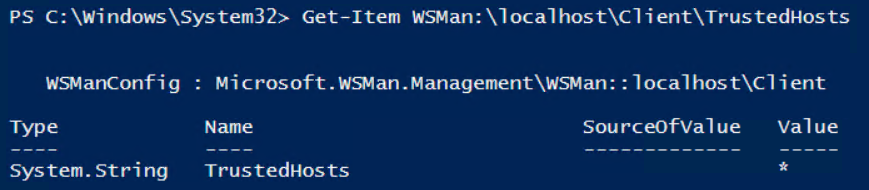
# On peut remplacer le paramètre \* par une liste de noms ou d'adresses IP qui sont séparés par des virgules.

**Restart-Service WinRM -Force**

# On redémarre le service WinRM pour s'assurer que les nouveaux paramètres sont utilisés.

**Get-Item WSMan:\localhost\Client\TrustedHosts**

# Cette commande affiche les ordinateurs auxquels on a confiance.



**Il faut redémarrer les deux serveurs.**

**Vous devez exécuter la prochaine commande PowerShell sur le SERVEUR1.**

La commande spécifie le nom de l'ordinateur source qui effectue un ping sur un ordinateur distant.

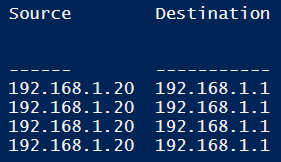
Il faut s'authentifier sur le SERVEUR2 parce qu'il est la source du PING.

**Test-Connection -Source 192.168.1.20 `**

**-ComputerName 192.168.1.1 `**

**-WsmanAuthentication Default `**

**-Credential Administrateur**

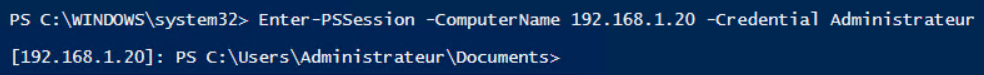


**Exemples pour exécuter des commandes PowerShell à distance**

**Vous devez exécuter les prochaines commandes PowerShell sur le SERVEUR1.**

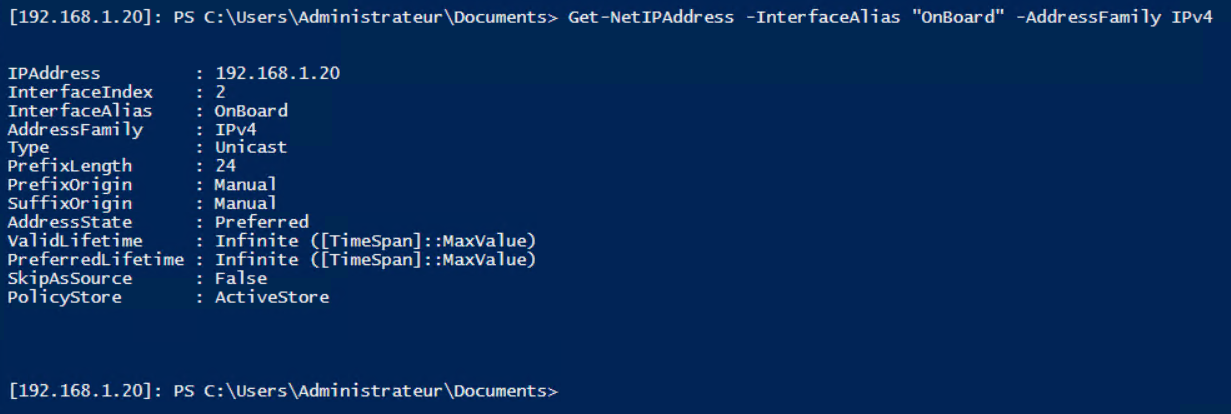
**Enter-PSSession -ComputerName 192.168.1.20 -Credential Administrateur**

**# La commande permet de se connecter au SERVEUR2.**



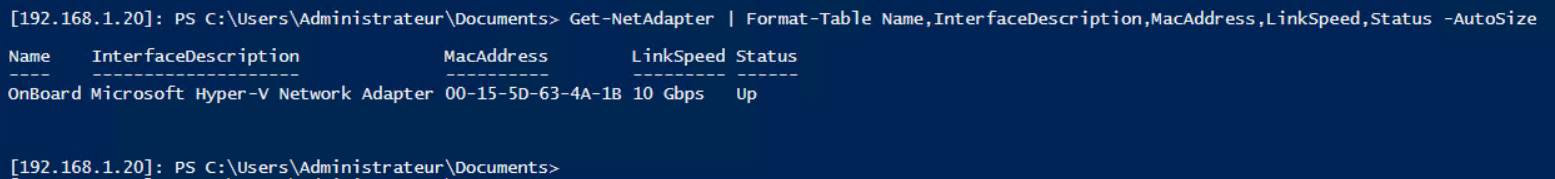
**Get-NetIPAddress -InterfaceAlias "Ethernet0" -AddressFamily IPv4**

**# La commande s'exécute sur le SERVEUR2 et non pas sur le SERVEUR1.**



**Get-NetAdapter | Format-Table Name,InterfaceDescription,MacAddress,LinkSpeed,Status -AutoSize**

**# La commande s'exécute sur le SERVEUR2 et non pas sur le SERVEUR1.**



**Exit-PSSession**

**# La commande permet de se déconnecter du SERVEUR2.**



**Exemple 1 - pour exécuter un script PowerShell à distance**

**Le code s'exécute sur le SERVEUR1 pour récupérer des informations sur le SERVEUR2.**

**Clear-Host**

**$IP = "192.168.1.20"**

**$cred = Get-Credential -Credential "Administrateur"**

**$code = { Get-NetIPAddress -InterfaceAlias "Ethernet0" -AddressFamily IPv4**

**Get-NetAdapter | Format-Table Name,InterfaceDescription,**

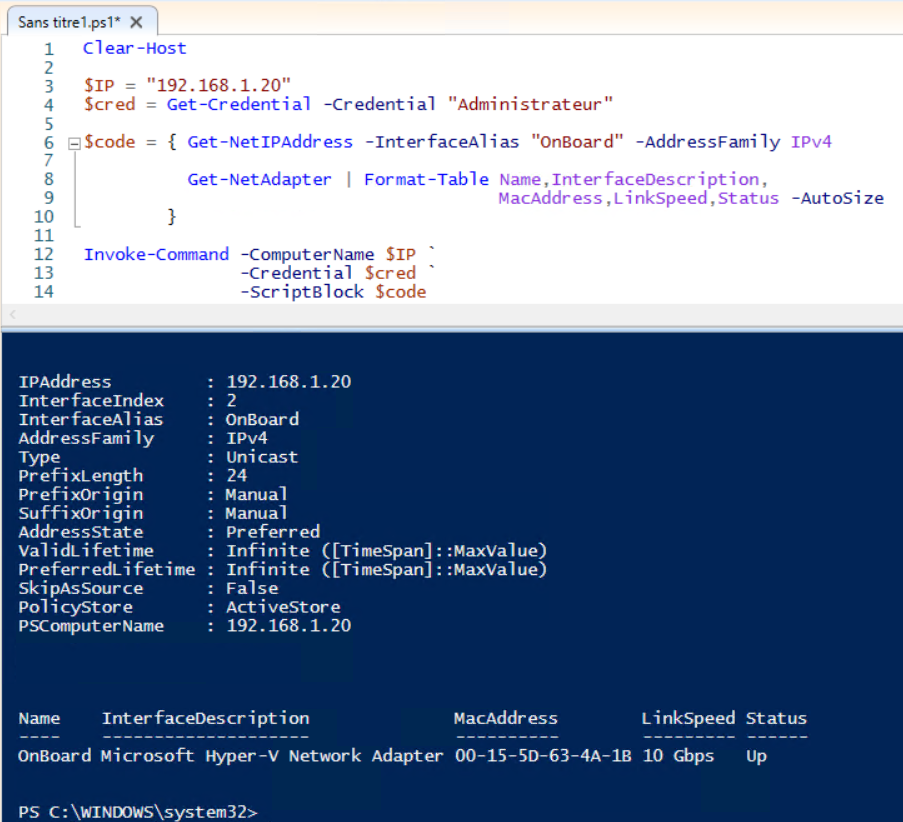
**MacAddress,LinkSpeed,Status -AutoSize**

**}**

**Invoke-Command -ComputerName $IP `**

**-Credential $cred `**

**-ScriptBlock $code**



**Exemple 2 - pour exécuter un script PowerShell à distance**

**Le code s'exécute sur le SERVEUR1 pour effectuer des modifications sur le SERVEUR2.**

* change le mot de passe de l'Administrateur
* s'assure que le mot de passe de l'Administrateur n'expire jamais

**Clear-Host**

**$IP = "192.168.1.20"**

**$code = { $mdp = convertTo-SecureString -AsplainText "Secret1" -Force**

**Set-LocalUser -Name Administrateur `**

**-Password $mdp `**

**-PasswordNeverExpires $true**

**}**

**Invoke-Command -ComputerName $IP `**

**-ScriptBlock $code**

Le paramètre **-Credential** de **Invoke-Command** n’est pas obligatoire si le mot de passe du compte Administrateur est le même sur les deux serveurs.