

# Tester une connexion en PowerShell avec Test-NetConnection

Dans ce tutoriel, je vais vous parler de la Cmdlet PowerShell `Test-NetConnection` qui va vous permettre de lancer des diagnostics réseaux.

Cette commande peut remplacer les utilitaires suivants :

- Ping
- Telnet
- Traceroute (tracert)

Avec une seule Cmdlet PowerShell vous allez pouvoir tout faire !

Dans ce tutoriel voici ce que l'on va voir :

- Comment faire un ping en PowerShell
- Comment faire un « telnet' » en PowerShell pour tester un port (TCP)
- Comment faire un trace route en PowerShell

Avant de commencer, la Cmdlet `Test-NetConnection` prend un premier paramètre `-ComputerName` qui n'est pas obligé d'être nommé.

## Faire un ping en PowerShell avec Test-NetConnection

On va commencer avec une utilisation simple, qui va être de faire un ping.

```
Test-NetConnection google.fr
```

image.png

Comme on peut le voir sur la capture, la commande nous retourne ces informations :

- Adresse ip distante résolu quand on test un nom de domaine
- L'interface et l'adresse IP source
- Le résultat du ping
- Le temps de latence

Maintenant, l'intérêt de PowerShell c'est que l'on va pouvoir faire des scripts, pour illustrer cela via un script simple qui nous permet de tester un ping.

```
<#  
[Script for TEST Connection  
#>  
  
param(  
[ $RemoteHost = "google.fr"  
)  
  
$ResultTest = Test-NetConnection -ComputerName $RemoteHost  
  
if( $ResultTest.PingSucceeded ){  
[Write-Output "Test OK for $RemoteHost"  
}else{  
  
[Write-Output "/!\ Test KO for $RemoteHost"  
}
```

Pour récupérer le résultat de la commande, il faut tester PingSucceeded.

Voici le script le résultat du script avec un hôte qui répond et qui ne répond pas.

image.png

Si vous souhaitez faire plusieurs ping ou un ping en continu, il faut utiliser la Cmdlet `Test-Connection`.

Pour effectuer 100 ping

```
Test-Connection google.fr -Count 100
```

Pour un ping en continu :

```
Test-Connection google.fr -Count ([int32]::MaxValue)
```

# Tester un port en PowerShell avec Test-NetConnection

Maintenant, on va voir comment tester un port en PowerShell comme on le ferait avec telnet.

L'avantage d'utiliser `Test-NetConnection` par rapport à telnet, c'est qu'il n'y a pas besoin d'installer le client Telnet.

Pour tester un port, il faut ajouter le paramètre `-Port` et le numéro de celui.

Ce qui nous donne :

```
Test-NetConnection rdr-it.com -Port 443
```

image.png

Ici, j'ai testé le port 443 (HTTPS) du serveur du site Internet, pour avoir le résultat du test, il faut regarder la valeur de `TcpTestSucceeded`.

En cas d'échec, la Cmdlet avec tester l'hôte à l'aide d'un ping.

image.png

On peut aussi exploiter ce retour dans un script PowerShell, à la différence du ping, il faut analyser le résultat de `TcpTestSucceeded`.

## Tester la route avec Test-NetConnection

Pour finir ce tutoriel, nous allons voir comment tester le routage avec `Test-NetConnection` en PowerShell.

Si vous avez plusieurs routes différentes de configurer, le premier paramètre que l'on peut utiliser est `-DiagnoseRouting`, qui va analysé quel route (passerelle) va être utiliser pour accéder au serveur distant.

image.png

Le second paramètre que l'on va voir est `-traceroute` qui permet d'avoir l'équivalent d'un `tracert` et de voir tous les routeurs qui sont utilisés.

image.png