

Nagios / NSclient++ - Utiliser des scripts powershell pour les checks



Difficulté : Confirmé

Notions : Supervision, nagios, agent, NSclient++, powershell.

S'applique à : Nagios / NS client ++ toutes versions

I.Introduction

Certaines informations ont besoins d'être monitorée mais ne sont accessibles que via des requêtes Powershell locales.

L'exécution de scripts doit être permise sur la machine. Pour cela utiliser la commande : **Set-executionPolicy Unrestricted**

II. Normalisation

Lors de l'écriture du script il est essentiel de respecter les points suivants :

- Nagios se base sur les Exit Codes. Il reconnaît les valeurs suivantes :
 - 0 : OK
 - 1 : Warning
 - 2 : Critical

Il faut donc prendre en compte lors de l'écriture du script que les exitcode doivent être générés et les 3 cas ci-dessus traités. Cela se fera en ajoutant la ligne suivante lors de la fin du traitement :

```
exit <code> # Ce code est un <Niveau>
```

par exemple :

```
exit 0 # This is the OK value
```

Si l'on veut pouvoir grapher ensuite les valeurs et les avoir correctement affichées dans le nagios, il faut respecter le formatage des données :

```
write-host "<Etat> : <intitulé valeur 1> : $Valeur1, <Intitulé valeur 2> : $Valeur2 |  
'<variablenagios1>'=$Valeur1 '<variablenagios2>'=$Valeur2"
```

Par exemple :

```
Write-Host "WARNING : Total Telnet connections : $TotalConnections, Active Telnet connections  
: $ActiveConnections | 'Nb_Total_Connections'=$TotalConnections  
'Nb_Active_Connections'=$ActiveConnections"
```

III. Déclarer le script sur NSClient++

Pour que le script soit utilisable, il faut le placer dans le répertoire : **C:\Program Files\NSClient++\scripts**

Puis il faut le déclarer dans à la fin du fichier **C:\Program Files\NSClient++\nsclient.ini**

nsclient.ini

```
[/settings/external scripts/scripts]  
<nom du script 1 (sans .ps1)> = cmd /c echo scripts\<script1>.ps1; exit($lastexitcode) |  
powershell.exe -command -  
<nom du script 2 (sans .ps1)> = cmd /c echo scripts\<script2>.ps1 $ARGS$;  
exit($lastexitcode) | powershell.exe -command -
```

Si le script prends des arguments en entrée, il faut également ajouter le bloc suivant avant le bloc ci-dessus :

```
[/settings/external scripts]  
allow arguments = True
```

Par exemple :

nsclient.ini

```
[/settings/external scripts]
allow arguments = True

[/settings/external scripts/scripts]
Telnet_Connections = cmd /c echo scripts\Telnet_Connections.ps1; exit($lastexitcode) |
powershell.exe -command -
Check_TelnetConnections = cmd /c echo scripts\Check_TelnetConnections.ps1 $ARGS$;
exit($lastexitcode) | powershell.exe -command -
```

IV. Tester le script

Sur le Nagios, l'on pourra tester le script avec la commande :

```
./check_nrpe -H <hostname ou ip> -c <nom de la commande> -a '<arguments>'
```

```
[a] [0] [1] $ ./check_nrpe -H [redacted] -c Check_TelnetConnections -a '-warning 380 -critical 400'
OK : Total Telnet connections : 294, Active Telnet connections : 255 |'Nb_Total_Connections'=294 'Nb_Active_Connections'=255
```

Revision #2

Created 2026-07-29 16:36:38 UTC by Olivier

Updated 2026-07-29 16:38:53 UTC by Olivier