

Partie 1 : maquette script_1.PS1 :

On commence par l'autorisation de scripts PowerShell

```
Set-ExecutionPolicy Unrestricted  
#auto exécution de scripts powershell  
#1 partie maquette
```

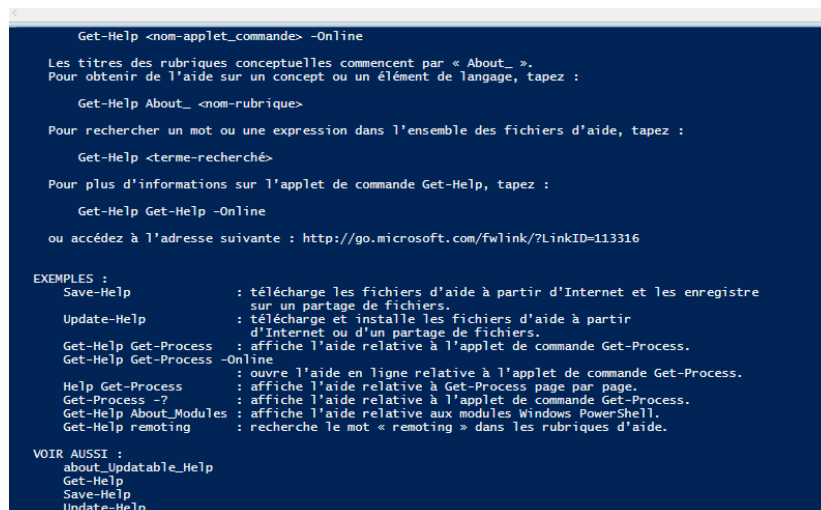
Partie 2 découverte :

script_2.PS1 :

CTRL+ESPACE comme Intellisense/les autres IDE(Eclipse) donne un aperçu de la tabulation.

La commande help est l'équivalent du man sous Linux

Get-Help <nom-applet_commande>



```
Get-Help <nom-applet_commande> -Online  
  
Les titres des rubriques conceptuelles commencent par « About_>.  
Pour obtenir de l'aide sur un concept ou un élément de langage, tapez :  
  
Get-Help About_ <nom-rubrique>  
  
Pour rechercher un mot ou une expression dans l'ensemble des fichiers d'aide, tapez :  
  
Get-Help <terme-recherché>  
  
Pour plus d'informations sur l'applet de commande Get-Help, tapez :  
  
Get-Help Get-Help -Online  
  
ou accédez à l'adresse suivante : http://go.microsoft.com/fwlink/?LinkID=113316  
  
EXEMPLES :  
Save-Help : télécharge les fichiers d'aide à partir d'Internet et les enregistre  
           sur un partage de fichiers.  
Update-Help : télécharge et installe les fichiers d'aide à partir  
             d'Internet ou d'un partage de fichiers.  
Get-Help Get-Process : affiche l'aide relative à l'applet de commande Get-Process.  
Get-Help Get-Process -Online : ouvre l'aide en ligne relative à l'applet de commande Get-Process.  
Help Get-Process : affiche l'aide relative à Get-Process page par page.  
Get-Process -? : affiche l'aide relative à l'applet de commande Get-Process.  
Get-Help About_Modules : affiche l'aide relative aux modules Windows PowerShell.  
Get-Help remoting : recherche le mot « remoting » dans les rubriques d'aide.  
  
VOIR AUSSI :  
about_Update-Help  
Get-Help  
Save-Help  
Update-Help
```

Par exemple Get-ITEM : ces paramètres sont aussi listés dans la partie commande

Commandes X

Modules: Tout Actualiser

Nom: get-ite

Get-Item
Get-ItemProperty
Get-ItemPropertyValue

Nom: Get-Item
Module: Microsoft.PowerShell.Management (Importé)

Paramètres pour « Get-Item » :

Path LiteralPath

Path: *

Credential:

Exclude:

Filter:

☐ Force

Include:

Stream:

☐ UseTransaction

Paramètres communs

Avec get-help

```
PS C:\Users\Administrateur> get-help Get-Item

NOM
    Get-Item

SYNTAX
    Get-Item [-Path] <string[]> [<CommonParameters>]
    Get-Item [<CommonParameters>]

ALIAS
    gi

REMARQUES
    Get-Help ne parvient pas à trouver les fichiers d'aide de cette applet de commande sur cet ordinateur. Il ne trouve qu'une aide partielle.
    -- Pour télécharger et installer les fichiers d'aide du module comportant cette applet de commande, utilisez Update-Help.
    -- Pour afficher en ligne la rubrique d'aide de cette applet de commande, tapez : « Get-Help Get-Item -Online » ou accédez à https://go.microsoft.com/fwlink/?LinkID=113319.
```

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\2Découverte> Get-Item ".\perlimpinpin.txt"

Répertoire: E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\2Découverte

Mode                LastWriteTime         Length Name
----                -
-a-----         19/03/2021    20:38             6 perlimpinpin.txt
```

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\2Découverte> Get-Item "."

Répertoire: E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres

Mode                LastWriteTime         Length Name
----                -
d-----         19/03/2021    20:37             2 2Découverte
```

Réponses :

```
$command_get_item_perlimpinpin= Get-Item "./perlimpinpin.txt"
```

Taille du fichier en octets

```
$command_get_item_perlimpinpin.Length
```

```
PS E:\mylaptop\Cdesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scriptsv2\autres\2Découverte> .\script_2_v4_one_command.PS1  
6
```

#Si Le fichier est en lecture seule

```
$command_get_item_perlimpinpin.IsReadOnly
```

```
False
```

L'heure du dernier accès au fichier

```
$command_get_item_perlimpinpin.LastAccessTime
```

ou

```
$command_get_item_perlimpinpin.LastAccessTimeUtc
```

```
jeudi 1 avril 2021 10:00:12  
jeudi 1 avril 2021 08:00:12
```

Partie 2 découverte :

script_3.PS1 :

```
Get-ChildItem ".\" | Where-Object -Property Length -GT 100
```

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\2Découverte>
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\2Découverte> Get-ChildItem ".\" | Where-Object -Property Length -GT 100

Répertoire : E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\2Découverte

Mode                LastWriteTime         Length Name
----                -
-a-----         20/03/2021   14:57             180 script_2.PS1
-a-----         20/03/2021   15:04             259 script_3.PS1
```

Partie 2 découverte :

script_4.PS1 :

```
Get-ChildItem ".\" | Where-Object -Property Length -GT 100 | Select-Object Name, Length
```

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\2Découverte> Get-ChildItem ".\" | Where-Object -Property Length -GT 100
```

Name	Length
script_2.PS1	195
script_2.v2.PS1	180
script_3.PS1	288

Partie 2 découverte :

script_5.PS1 :

```
$ifIndex_value=Get-NetIPAddress -InterfaceAlias "Ethernet" -AddressFamily "IPv4" | Select-Object -Property ifIndex | ForEach-Object -MemberName ifIndex  
echo "L'index de la carte Ethernet est : $ifIndex_value"
```

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 -  
echo "L'index de la carte Ethernet est : $ifIndex_value"  
L'index de la carte Ethernet est : 18
```

Notes : L'index vaut 11 sur le windows Server.

Partie 2 découverte :

script_6.PS1 :

Résultat prévu : 3920 octets

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\2Découverte> ls
```

Répertoire : E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\2Découverte

Mode	LastWriteTime	Length	Name
-a----	19/03/2021 20:38	6	perlimpinpin.txt
-a----	20/03/2021 15:11	195	script_2.PS1
-a----	20/03/2021 15:24	755	script_2.v2.PS1
-a----	20/03/2021 15:13	1046	script_3.PS1
-a----	20/03/2021 15:18	256	script_4.PS1
-a----	20/03/2021 17:16	1141	script_5.PS1
-a----	20/03/2021 17:29	277	script_6.PS1
-a----	20/03/2021 17:35	244	script_6.v2.PS1

Calculatrice

Standard

3676 + 244 =

3 920

Résultats obtenus :

Sous forme complète

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\2Découverte> Get-Childitem ".\" | measure-object -Property length -Sum
```

```
Count      : 8  
Average    :  
Sum        : 3920  
Maximum    :  
Minimum    :  
Property   : Length
```

Filtrer encore mieux sur une ligne :

```
Get-ChildItem ".\" | measure-object -Property length -Sum | Select-Object -  
Property Sum | ForEach-Object -MemberName Sum
```

3916

Partie 3 Statistiques de RTT :

script_7.PS1 :

```
$proxy_url="proxya.u-pec.fr"
$proxy_url="8.8.8.8"

$counting_limit="10"
Test-Connection $proxy_url -count $counting_limit
```

Source	Destination	IPv4Address	IPv6Address	Bytes	Time(ms)
RT107-32	8.8.8.8	8.8.8.8	2001:4860:4860::8888	32	28
RT107-32	8.8.8.8	8.8.8.8	2001:4860:4860::8888	32	21
RT107-32	8.8.8.8	8.8.8.8	2001:4860:4860::8888	32	7
RT107-32	8.8.8.8	8.8.8.8	2001:4860:4860::8888	32	17
RT107-32	8.8.8.8	8.8.8.8	2001:4860:4860::8888	32	9
RT107-32	8.8.8.8	8.8.8.8	2001:4860:4860::8888	32	9
RT107-32	8.8.8.8	8.8.8.8	2001:4860:4860::8888	32	9
RT107-32	8.8.8.8	8.8.8.8	2001:4860:4860::8888	32	12
RT107-32	8.8.8.8	8.8.8.8	2001:4860:4860::8888	32	8
RT107-32	8.8.8.8	8.8.8.8	2001:4860:4860::8888	32	27

Partie 3 Statistiques_RTT :

script_8.PS1 :

La colonne ResponseTime donne le temps d'un RTT/d'aller-retour de chaque test de connexion.

```
$proxy_url="proxya.u-pec.fr"
$proxy_url="8.8.8.8"

$counting_limit="10"

Test-Connection $proxy_url -count $counting_limit | Get-Member -View All
```

Qualifiers	Property	System.Management.QualifierDataCollection Qualifiers {get;}
RecordRoute	Property	uint32 RecordRoute {get;set;}
ReplyInconsistency	Property	bool ReplyInconsistency {get;set;}
ReplySize	Property	uint32 ReplySize {get;set;}
ResolveAddressNames	Property	bool ResolveAddressNames {get;set;}
ResponseTime	Property	uint32 ResponseTime {get;set;}
ResponseTimeToLive	Property	uint32 ResponseTimeToLive {get;set;}
RouteRecord	Property	string[] RouteRecord {get;set;}
RouteRecordResolved	Property	string[] RouteRecordResolved {get;set;}
Scope	Property	System.Management.ManagementScope Scope {get;set;}
Site	Property	System.ComponentModel.ISite Site {get;set;}
SourceRoute	Property	string SourceRoute {get;set;}
SourceRouteType	Property	uint32 SourceRouteType {get;set;}
StatusCode	Property	uint32 StatusCode {get;set;}

Ou juste Avec Get-Member :

```
Test-connection 8.8.8.8 |Get-Member
```

[Capturing the PingReplyDetails \(RTT\) value from Test-NetConnection : PowerShell \(reddit.com\).](#)
[Test-NetConnection vs. Test-Connection - Testing a network connection with PowerShell | 4sysops](#)

Partie 3 Statistiques_RTT :

script_9_v5_testconnection_debug.PS1 :

```
$proxy_url="proxya.u-pec.fr"
$proxy_url="8.8.8.8"
$counting_limit="10"
$command = (Test-Connection $proxy_url -count $counting_limit) | measure-
object -Property ResponseTime -Average
$average_RTT=$command.Average

Write-Host "For" $counting_limit "pings, the average RTT is of" $average_RTT
```

```
For 10 pings, the average RTT is of 10,6
```


Partie 4 Résolution DNS :

script_10.PS1 :

```
$url="www.google.com"
```

```
Resolve-DnsName -Name $url
```

Name	Type	TTL	Section	IPAddress
www.google.com	AAAA	114	Answer	2a00:1450:4007:810::2004
www.google.com	A	15	Answer	216.58.198.196

La propriété qui donne le type d'enregistrement est le champ type

partie 4 Résolution DNS :

script_11.PS1 :

```
$url="www.google.com"
```

```
Resolve-DnsName -Name $url | Where-Object -Property "Type" -eq "A"
```

Name	Type	TTL	Section	IPAddress
www.google.com	A	154	Answer	142.250.179.100

Partie 4 Résolution DNS :

script_12.PS1 :

```
$url="www.google.com"
```

```
$site_name="Google"
```

```
$IP_value=Resolve-DnsName -Name $url | Where-Object -Property "Type" -  
eq "A" | Select-Object -Property IPAddress | ForEach-Object -  
MemberName IPAddress  
echo "L'adresse IPv4 de $site_name est : $IP_value"
```

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\2Découverte> $url="www.google.com"  
$site_name="Google"  
$IP_value=Resolve-DnsName -Name $url | Where-Object -Property "Type" -eq "A" | Select-Object -Property IPAddress | ForEach-Object -MemberName IPAddress  
echo "L'adresse IPv4 de $site_name est : $IP_value"  
L'adresse IPv4 de Google est : 216.58.206.228
```

Partie 4 Résolution DNS : fichier lié `hosts.txt`

`script_13.ps1` : `dns.ps1`

```
$hostfile="./hosts.txt"
```

```
#boucle foreach
```

```
foreach($line in Get-Content $hostfile) {  
    $IP_value= Resolve-DnsName -Name $line | Where-Object -Property "Type" -  
eq "A" | Select-Object -Property IPAddress | ForEach-Object -  
MemberName IPAddress
```

```
    Write-Host "$line $IP_value"
```

```
}
```

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL/scripts/autres\4Résolution_DNS> get-content .\hosts.txt | select-object -unique  
www.google.com  
www.facebook.com  
  
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL/scripts/autres\4Résolution_DNS> $hostfile="./hosts.txt"  
  
#boucle foreach  
foreach($line in Get-Content $hostfile) {  
    $IP_value= Resolve-DnsName -Name $line | Where-Object -Property "type" -eq "A" | Select-object -Property IPAddress | ForEach-object -MemberName IPAddress  
    echo "$line $IP_value"  
}  
www.google.com 142.250.74.228  
www.facebook.com 179.60.192.36
```

Partie 5. Détecteur de coupure de connexion :

script_14.PS1 :
Partie sauté

```
5Détecteur_de_coupure_de_connexion > > script_14.PS1
1 while (1) {
2     # ping 192.168.1.254 -n 1 -w 2
3     ping proxya.u-pec.fr -n 1 -w 2 | Out-Null
4
5     # Write-Host -Nonewline $? #aurait fournit la valeur de retour
6
7     if ($? -eq "True") {
8         Write-Host -Nonewline "."
9     }
10    else {
11        Write-Host -Nonewline "!"
12        [System.Console]::Beep();
13    }
14    Start-Sleep 10 #à placer ici
15 }
```

```
#-n 1 maximum 1 requete au lieu de -c
#-w 2 sec en timeout
# > /dev/null | Out-Null
# > /dev/null ; > NUL on cmd
# True au lieu de $? -eq 0
```

```
PS C:\Users\Administrateur.WIN-UU9EAKBJJH3\Desktop> C:\Users\Administrateur.WIN-UU9EAKBJJH3\Desktop\
..!!!!!!..!!!!!!
```

Partie 5. Détecteur de coupure de connexion :

script_15.PS1 :

```
$route=$(Get-NetRoute | Where-Object -Property "RouteMetric" -eq "0" | Select-
Object -Property NextHop | ForEach-Object -MemberName NextHop)
$ip=$args[0]
if (!$route) {
    echo "Ce PC n'a pas de passerelle par défaut !"
    exit 1
}

echo "L'adresse passerelle est :" $route

if (!$args[0]) {
    while (1){
        ping $route -n 1 -w 2 | Out-Null

        if ( $? -eq 0) {
            Write-Host -Nonewline "!"
        }
    }
}
```

```

    }
    else{
        Write-Host -Nonewline "."
    }

    Start-Sleep 10
}
}
echo "L'adresse IP en argument est :" $ip
while (1){

    ping $ip -n 1 -w 2 | Out-Null

    if ($? -eq 0) {
        Write-Host -Nonewline "!"
    }
    else{
        Write-Host -Nonewline "."
    }

    Start-Sleep 10
}

```

- Cas 1 : Avec le ping vers la passerelle par défaut :

```

# output :
L'adresse passerelle est :
10.0.2.2
.....!.!!!.

```

- Cas 2 : Sans passerelle par défaut :

```

PS C:\Users\Administrateur.WIN-UU9EAKBJJH3\Desktop> C:\Users\Administrateur.WIN-UU9EAKBJJH3\Desktop\
Ce PC n'a pas de passerelle par défaut !

```

- Cas 3 : Avec adresse IP en argument

```

PS C:\Users\Administrateur.WIN-UU9EAKBJJH3\Desktop> & '.\Sans titre1.ps1' 10.0.2.1
L'adresse passerelle est :
10.0.2.2
L'adresse IP en argument est :
10.0.2.1
!.!!!!!!

```

Partie 6. Temps de réponse de sites Web :

script_16.PS1 :

```
$hostfile="./hosts_url.txt"

#boucle foreach
foreach($line in Get-Content $hostfile) {
    $URL_content_value= Invoke-WebRequest -Uri $line -UseBasicParsing
    echo "$line output : "
    echo "$URL_content_value"
}
```

Avec qu'un seul URL car beaucoup d'URL, le résultat est trop long et illisible :

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration
système\TP3 POWERSHELL\scripts\autres\4Résolution_DNS> Invoke-WebRequest -Uri
"http://www.google.com" -UseBasicParsing

StatusCode      : 200
StatusDescription : OK
Content         : <!doctype html><html itemscope=""
itemtype="http://schema.org/WebPage" lang="fr"><head><meta content="text/html;
charset=UTF-8"
http-equiv="Content-Type"><meta
content="/logos/doodles/2021/spring-2021...
RawContent      : HTTP/1.1 200 OK
                  X-XSS-Protection: 0
                  X-Frame-Options: SAMEORIGIN
                  Vary: Accept-Encoding
                  Transfer-Encoding: chunked
                  Accept-Ranges: none
                  Cache-Control: private, max-age=0
                  Content-Type: text/html; c...
Forms           :
Headers         : {[X-XSS-Protection, 0], [X-Frame-Options, SAMEORIGIN], [Vary,
Accept-Encoding], [Transfer-Encoding, chunked]...}
Images          : {}
InputFields     : {}
Links           : {@{outerHTML=<a class="gbzt gbz01 gbp1" id=gb_1
href="https://www.google.fr/webhp?tab=ww"><span class=gbtb2></span><span
class=gbts>Recherche</span></a>; tagName=A; class=gbzt gbz01
gbp1; id=gb_1; href=https://www.google.fr/webhp?tab=ww}, @{outerHTML=<a class=gbzt
id=gb_2
href="http://www.google.fr/imghp?hl=fr&tab=wi"><span
class=gbtb2></span><span class=gbts>Images</span></a>; tagName=A; class=gbzt;
id=gb_2;
href=http://www.google.fr/imghp?hl=fr&tab=wi}, @{outerHTML=<a
class=gbzt id=gb_8 href="http://maps.google.fr/maps?hl=fr&tab=w1"><span
class=gbtb2></span><span class=gbts>Maps</span></a>; tagName=A;
class=gbzt; id=gb_8; href=http://maps.google.fr/maps?hl=fr&tab=w1}, @{outerHTML=<a
class=gbzt id=gb_78
href="https://play.google.com/?hl=fr&tab=w8"><span class=gbtb2></span><span
class=gbts>Play</span></a>; tagName=A; class=gbzt;
id=gb_78; href=https://play.google.com/?hl=fr&tab=w8}...}
ParsedHtml      :
RawContentLength : 49178
```

Partie 6 Temps de réponse de sites Web :

script_17.PS1 :

- Sans formatage en une ligne de commande

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\6Temps_de_réponse_de_sites_web> $host_url="http://quelleheureestilenjoy.com/00.mp4"

Measure-Command {Invoke-WebRequest -Uri $host_url -UseBasicParsing }
```

```
Days           : 0
Hours          : 0
Minutes        : 0
Seconds        : 4
Milliseconds    : 427
Ticks          : 44270275
TotalDays      : 5,1238744212963E-05
TotalHours     : 0,00122972986111111
TotalMinutes   : 0,0737837916666667
TotalSeconds   : 4,4270275
TotalMilliseconds : 4427,0275
```

- Avec un meilleur formatage :

```
$host_url="http://quelleheureestilenjoy.com/00.mp4"

$time_to_dl=Measure-Command {Invoke-WebRequest -Uri $host_url -
UseBasicParsing } | Select-Object -Property Milliseconds | ForEach-Object -
MemberName Milliseconds

echo "Temp pour download/lire la vidéo d'EnjoyPhoenix : $time_to_dl en ms "
```

```
PS E:\mylaptop\CDesktop\offline\augustin mojarad ECUE 31 - Administration système\TP3 POWERSHELL\scripts\autres\6Temps_de_réponse_de_sites_web> $host_url="http://quelleheu

$time_to_dl=Measure-Command {Invoke-WebRequest -uri $host_url -UseBasicParsing } | Select-Object -Property Milliseconds | ForEach-Object -MemberName Milliseconds

echo "Temp pour download/lire la vidéo d'EnjoyPhoenix : $time_to_dl en ms "
Temp pour download/lire la vidéo d'EnjoyPhoenix : 432 en ms
```

[3 ways to measure your Powershell script's speed | Pluralsight](#)

Partie 6 Temps de réponse de sites web :

script_18.PS1 :

```
6Temps_de_réponse_de_sites_Web > script_18.PS1
1 $proxyfile="./proxy.txt"
2 # list
3 # or multi list
4 #with two separated lists
5 $proxys_array=@('')
6 $access_time_array=@('')
7
8
9
10 #boucle foreach
11 foreach($line in Get-Content $proxyfile) {
12     $time_to_access=Measure-Command {Invoke-WebRequest -Uri $line -UseBasicParsing } | Select-Object -Property Milliseconds | \
13     ForEach-Object -MemberName Milliseconds
14 # append to list
15 Write-Output "Temps pour accéder au proxy $line : $time_to_access en ms "
16 $proxys_array += $proxys_array+$line
17 $access_time_array += $access_time_array + $time_to_access
18
19 }
20
21 # order list by the min value show the multi list proxy value
22 # Write-Output $access_time_array | sort-Object | Select-Object -Unique
23 # $shortest_access_time=Write-Output $access_time_array | sort-Object | Select-Object -First 1
24 # $shortest_access_time=Write-Output $access_time_array | sort-Object -Descending
25 # $shortest_access_time=Write-Output $access_time_array | sort-Object |
26 $minvalue=[int]($access_time_array | measure -Minimum).Minimum
27 $shortest_access_time = $access_time_array.IndexOf($minvalue)
28 $shortest_proxy= $proxys_array[$shortest_access_time]
29 $shortest_access_time= $access_time_array[$shortest_access_time]
30
31 # |Where-Object -Value -gt
32 Write-Host -NoNewLine "shortest time : "
33 Write-Host $shortest_access_time " ms"
34
35 Write-Host -NoNewLine "shortest proxy : "
36 Write-Host $shortest_proxy
37
38 # |Where-Object -Value -gt
39 Write-Host -NoNewLine "shortest time : "
40 Write-Host $shortest_access_time " ms"
41
42 Write-Host -NoNewLine "shortest proxy : "
43 Write-Host $shortest_proxy
44
45 Invoke-WebRequest : Le serveur distant a retourné une erreur : (404) Introuvable.
46 Au caractère Ligne:12 : 40
47 + ... asure-Command {Invoke-WebRequest -Uri $line -UseBasicParsing } | S ...
48 +
49 + CategoryInfo          : InvalidOperation : (System.Net.HttpWebRequest:HttpWebRequest) [Invoke-WebRequest], WebException
50 + FullyQualifiedErrorId : webcmdletWebResponseException,Microsoft.PowerShell.Commands.InvokeWebRequestCommand
51
52 Temps pour accéder au proxy 47.52.222.165 : 601 en ms
53 Invoke-WebRequest : Impossible de se connecter au serveur distant
54 Au caractère Ligne:12 : 40
55 + ... asure-Command {Invoke-WebRequest -Uri $line -UseBasicParsing } | S ...
56 +
57 + CategoryInfo          : InvalidOperation : (System.Net.HttpWebRequest:HttpWebRequest) [Invoke-WebRequest], WebException
58 + FullyQualifiedErrorId : webcmdletWebResponseException,Microsoft.PowerShell.Commands.InvokeWebRequestCommand
59
60 Temps pour accéder au proxy 121.8.98.197 : 58 en ms
61 shortest time : 58 ms
62 shortest proxy : 121.8.98.197
```

Notes : Fastest_proxy au lieu de shortest proxy

Partie 7. Cartes réseau :

script_19.PS1 : Fichier lié [nicdetector.ps1](#) : (network interface card detector)

```
$command=Get-NetAdapter -Physical | Select-Object -
Property Name, Status,ifindex | Where-Object -Property "Status" -eq "Up"
$command_count_value = $command | Measure-Object -Property ifIndex -
Sum | ForEach-Object -MemberName Count
```

```
# case 2
if ($command_count_value -lt 1){
    Write-
Host "Pas d'interface branchée ! Vous devez en câbler au moins une."
}
# case 3
elseif ($command_count_value -gt 1){
    Write-
Host "Il y a plusieurs interfaces branchées ! Une seule carte doit être câblée
."
}
# case 1
elseif ($command | Where-Object -Property "Status" -eq "Up" ){
    $name_card=$command | Select-Object -Property Name | ForEach-Object -
MemberName Name
    Write-Host "Interface branchée : " $name_card
}
```

- Cas 1 :

```
PS C:\Users\Administrateur.WIN-UU9EAKBJJH3\Desktop> C:\Users\Administrateur.WIN-UU9EAKBJJH3\Desktop\
Interface branch  e : Ethernet
```

- Cas 2 :

```
PS C:\Users\Administrateur.WIN-UU9EAKBJJH3\Desktop> C:\Users\Administrateur.WIN-UU9EAKBJJH3\Desktop\
Pas d'interface branch  e ! Vous devez en câ bler au moins une.
```

- Cas 3 :

```
PS C:\Users\Administrateur.WIN-UU9EAKBJJH3> C:\Users\Administrateur.WIN-UU9EAKBJJH3\De
Il y a plusieurs interfaces branch  es ! Une seule carte doit   tre câ bl  e.
```

Notes : Les caractères accentués sont mal interprétés sur la VM Windows Server ce qui n'est pas le cas, des autres PC Windows 10 (Consumer)

Github :

[amineAUPEC/TP3-POWERSHELL: TP3-POWERSHELL IUT CRETEIL-VITRY LICENCE PRO ASUR \(github.com\)](https://github.com/amineAUPEC/TP3-POWERSHELL)

<https://github.com/amineAUPEC/TP3-POWERSHELL>

Annexes :

Partie 2 découverte :

script_2.PS1 :

Avec Select-object :

```
# Taille du fichier en octets  
Get-Item "./perlimpinpin.txt" | Select-Object -Property Length
```

```
Length  
-----  
6
```

Avec where-Object

```
#Si le fichier est en lecture seule  
Get-Item "./perlimpinpin.txt" | Where-Object IsReadOnly
```

Aucune sortie :

```
PS E:\mylaptop\CDesktop\offline\august
```

Avec Select-object :

```
# L'heure du dernier accès au fichier  
Get-Item "./perlimpinpin.txt" | Select-Object -Property LastAccessTime
```

```
LastAccessTime  
-----  
20/03/2021 15:14:22
```

Partie 3 Statistiques_RTT :

Avec Test-Net-Connection :

script_7_v2.PS1 :

```
$proxy_url="proxya.u-pec.fr"
$proxy_url="8.8.8.8"

$counting_limit="10"

foreach ($i in 1..$counting_limit){
    # Write-Host $i
    Test-NetConnection $proxy_url
}
```

Ping/ICMP Test, Waiting for echo reply.

```
foreach ($i in 1..$counting_limit){
    # Write-Host $i
    Test-NetConnection $proxy_url
}

ComputerName      : 8.8.8.8
RemoteAddress     : 8.8.8.8
InterfaceAlias    : Wi-Fi2
SourceAddress     : 192.168.1.119
PingSucceeded     : True
PingReplyDetails (RTT) : 9 ms

ComputerName      : 8.8.8.8
RemoteAddress     : 8.8.8.8
InterfaceAlias    : Wi-Fi2
SourceAddress     : 192.168.1.119
PingSucceeded     : True
PingReplyDetails (RTT) : 27 ms

ComputerName      : 8.8.8.8
RemoteAddress     : 8.8.8.8
InterfaceAlias    : Wi-Fi2
SourceAddress     : 192.168.1.119
PingSucceeded     : True
PingReplyDetails (RTT) : 39 ms
```

Partie 3 Statistiques_RTT :

Avec Test-Net-Connection :

script_8_v2.PS1 :

La colonne PingReplyDetails puis RoundtripTime RTT donne le temps d'un aller-retour de chaque test de connexion.

[Test-NetConnection \(NetTCPIP\) | Microsoft Docs](#)

```
$proxy_url="proxya.u-pec.fr"
```

```
$proxy_url="8.8.8.8"
(Test-NetConnection $proxy_url).pingreplydetails
# (tnc $proxy_url).pingreplydetails
# tnc $proxy_url | Get-Member
```

```
Status      : Success
Address     : 8.8.8.8
RoundtripTime : 22
Options     : System.Net.NetworkInformation.PingOptions
Buffer      : {97, 98, 99, 100...}
```

Avec Get-Member :

```
tnc 8.8.8.8 - | Get-Member
```

[Capturing the PingReplyDetails \(RTT\) value from Test-NetConnection : PowerShell \(reddit.com\).](#)
[Test-NetConnection vs. Test-Connection - Testing a network connection with PowerShell | 4sysops](#)

Partie 3 Statistiques_RTT :

Avec Test-Net-Connection :
script_9_v2.PS1 :

```
$proxy_url="proxya.u-pec.fr"
$proxy_url="8.8.8.8"
$counting_limit="10"
$command = (Test-NetConnection $proxy_url).PingReplyDetails.RoundTripTime
$counting_RTT="0"
foreach ($i in 1..$counting_limit){
    # Write-Host $i
    # Write-Host "command_value" $command
    # Write-Host "Previous value " $counting_RTT
    $counting_RTT= $counting_RTT + $command -as [int]
    # Write-Host $counting_RTT
    # (Test-
NetConnection $proxy_url).PingReplyDetails.RoundTripTime | measure-object -
Property Length -Sum
}
Write-Host $counting_RTT
```

90