

## Komendy:

- Write-Host
- Get-Process
- Get-Service
- Start-Service
- Get-EventLog
  - -LogName
- Get-WmiObject
  - Class (np. win32\_physicalmemory)
- Get-CimInstance
  - -ClassName
  - -CimSession
- New-CimSessionOption (w zmiennej)
  - -Protocol
- New-CimSession (w zmiennej)
  - -ComputerName
  - -SessionOption
- Enable-WSManCredSSP -Role {Server/Client}
- Get-WSManCredSSP
- Invoke-Command
  - -ComputerName {np. localhost}
  - -Credential {\$cred}
  - -ScriptBlock {}
- Get-job
- Receive-Job
  - -Id {id}
- Remove-Job
  - -Id {id}
- New-PSDrive
  - -Name {name}
  - -PSProvider FileSystem
  - -Root {dir}
- New-Item
  - -Path {path}
  - -ItemType File/Directory

- -Force
- Copy-Item
  - -Path {katalog}
  - -Destination {katalog}
  - -ToSession {session}
- Compress-Archive
  - -Path {folder}
  - -DestinationPath {folder.zip}
- \$x = New-ScheduledTaskTrigger (do zmiennej)
  - -Daily
  - -At 15:00
- Register-ScheduledJob
  - -Name "job"
  - -Trigger \$trigger
  - -ScriptBlock {[skrypt]}
- Start-ScheduledTask
  - -TaskName {name}
- Start-Sleep
  - -Seconds {s}
- Get-ScheduledTask
- Unregister-ScheduledTask
  - -TaskName {nazwa}
- Get-Credential
  - -Message "Enter Password"
  - -UserName (np. "\$env:COMPUTERNAME\Administrator")
- Enable-LocalUser
  - -Name "Administrator"

#### Formatowanie:

- FW - Format Wide
- FT - Format Table
- FL - Format List
- Sort
- Select-Object

- Get-Member

Logika:

- If

```
if(wyr logiczne) {  
  
}  
    ○ -eq - equal  
    ○ -ne - not equal  
    ○ -gt greater than  
    ○ -ge greater than or equal  
    ○ -lt less than  
    ○ -le less than or equal
```
- foreach (\$x in {zakres}/{start..end}){  
 }

Inne:

- \$admCred = {smh}
- Enable-PSRemoting -Force - włącza sesje zadne
- Set-Item WSMAN:\localhost\client\trustedhosts -Value "\*" -Force -  
 pozwolenie na połączenia (Serwer)
- Set-ExecutionPolicy RemoteSigned - pozwala na uruchomienie skryptów
- \$env:{nazwa} - zmienne environment
- (Get-Service {nazwa}).Status - wyświetla status
- \$x.IsReadOnly = \$false/\$true
- \$x.Decrypt()
- \$x.Encrypt()



Przygotowanie przed testem.

Wewnątrz maszyny wirtualnej:

1. uruchom `Start-Transcript C:\imienazwisko_ppk1_grnr.txt`
2. utwórz plik `C:\imienazwisko_rpk1_grnr.txt` w którym umieścisz odpowiedzi do zadań.

### Test:

1. Posortuj listę procesów wg „TotalProcessorTime” (właściwość) które odwołuje się do pełnej nazwy właściwości.

```
Get-Process | sort TotalProcessorTime | ft ProcessName, TotalProcessorTime
```

2. Wyświetl tylko unikalne „Source” występujące wśród ostatnich 4 zdarzeń.

```
Get-EventLog -LogName System -Newest 4 | Select-Object -Property Source - Unique
```

3. Podaj polecenie który ma zapisać raport do pliku C:\Users\proc\_raport.txt o procesach uruchomionych w danej chwili i ilości wykorzystywanej przez nie pamięci, plik nie ma być nadpisywany a kolejne porcje informacji przy każdym uruchomieniach mają być dopisywane na koniec pliku, aby zapisy się nie wymieszały na początku skryptu przed informacjami o procesach ma znaleźć się data.

```
Get-Date|Out-File -FilePath C:\Users\proc_raport.txt -Append ; Get-Process | ft name, vm, pm |Out-File -FilePath C:\Users\raport_proc.txt -Append
```

4. Korzystając z metod WMI: Wyświetl dla każdego obiektu Win32\_NetworkAdapter tylko właściwość MACAddress oraz AdapterType.

```
Get-WmiObject -Class Win32_NetworkAdapter | ft macaddress, adaptertype
```

5. Korzystając z metod WMI i CIM: Korzystając z klasy Win32\_UserProfile wyświetl informacji o wszystkich profilach użytkowników na komputerze lokalnym

```
Get-WmiObject -Class Win32_UserProfile; Get-CimInstance Win32_UserProfile
```

6. Utwórz obiekt sesji oparty o protokół DCOM. Utwórz obiekt sesji do wybranego komputera z wykorzystaniem utworzonego obiektu sesji. Korzystając z obiektu sesji wyświetl informacje o pulpitych dostępnych na zdalnym komputerze.

```
$opt=New-CimSessionOption -Protocol DCOM
$s=New-CimSession -ComputerName [nazwaPC/$env:COMPUTERNAME] -
SessionOption $opt
Get-CimInstance -ClassName Win32_Desktop -CimSession $s
```

7. Napisz polecenie, które uruchomi usługę tylko o ile aktualnie ta usługa nie jest uruchomiona.

Przed i po uruchomieniu usługi dodaj polecenia wyświetlające na ekranie dodatkowe komunikaty. Dodaj do poprzedniej instrukcji polecenie else, które w przypadku, gdy usługa już działa wyświetli komunikat „Service is already running”

```
if( (Get-Service bits).Status -eq "Stopped") {
Write-Host "Starting service"
Start-Service bits
Write-Host "Service started"
}
else {
Write-Host "Service is already running"
}
```

8. Utwórz kilka plików w wybranym katalogu (tutaj c:\kosz). Zadeklaruj zmienną \$files i przypisz do niej wynik polecenia Get-ChildItem listującego zawartość folderu C:\kosz Napisz pętlę foreach, która dla każdego pliku z kolekcji \$files:

- a. Zaszyfruj ten plik (wywołaj metodę Encrypt() dla bieżącego elementu kolekcji)
- b. Zmień atrybut ReadOnly na \$true. Wykorzystaj w tym celu właściwość ReadOnly bieżącego elementu kolekcji. Sprawdź, czy pliki są zaszyfrowane i mają ustawiony atrybut tylko do odczytu (skorzystaj np. z eksploratora Windows). Napisz kolejną pętlę foreach, która: odszyfruje pliki i wygasi atrybut tylko do odczytu.

```
a)
$files = (Get-ChildItem c:\kosz)

foreach($X in $files){
$X.Encrypt()
}
```

```
b)
$files = (Get-ChildItem c:\kosz)
```

```
foreach($X in $files){  
$x.IsReadOnly = $false  
$x.Decrypt()  
}
```

9. Uruchom na systemie zdalnym polecenie wyświetlające nazwę komputera w sposób wsadowy (nieinteraktywny) korzystając z polecenia Invoke-Command

```
Invoke-Command -ComputerName {np. localhost} -ScriptBlock {  
$env:COMPUTERNAME }
```

10. Skonfiguruj komputer do przekazywania poświadczeń:

- a. Jako klient do przekazywania poświadczeń do komputera zdalnego.

```
Enable-PSRemoting -Force  
Enable-PSRemoting -SkipNetworkProfileCheck  
Set-Item WSMan:\localhost\Client\TrustedHosts -Value "*" -Force
```

- b. Jako serwer do przyjmowania poświadczeń Sprawdź bieżącą konfigurację CredSSP.

```
Enable-WSManCredSSP -Role Server  
Get-WSManCredSSP
```

11. Poniżej znajduje się lista czynności do wykonania w PowerShell

- Utwórz profil użytkownika PowerShell
- Edytuj plik profile.ps1 i dodaj polecenie utwórz wirtualny napęd TEMP: zmapowany do katalogu **c:\temp**
- Upewnij się, że execution policy pozwala na uruchamianie skryptów.

```
New-Item -Path $PROFILE -ItemType File  
(w pliku)> New-PSDrive -Name TEMP -PSProvider FileSystem -Root  
C:\temp  
Set-ExecutionPolicy RemoteSigned
```

12. Poniżej znajduje się lista czynności do wykonania w PowerShell

- a. Pobierz listę procesów i zapisz wynik w pliku

```
Get-Process | Out-File -FilePath {ścieżka_do_pliku}
```

- b. Skopiuj plik z jednego katalogu do drugiego

```
Copy-Item -Path "C:\ścieżka_do_katalogu_źródłowego" -Destination  
"C:\ścieżka_do_docelowego_katalogu"
```

- c. Skompresuj folder

```
Compress-Archive -Path {folder} -DestinationPath {folder.zip}
```

- d. Pobierz informacje o procesorze

```
Get-WmiObject -Class win32_processor
```

- e. Uruchom Get-Service na komputerze zdalnym, podaj rzeczywistą nazwę komputera

```
Invoke-Command -ComputerName $env:COMPUTERNAME -ScriptBlock { Get-  
Service }
```

- f. Wyświetl listę jobów

```
Get-Job
```

- g. Pobierz wynik job-a zdefiniowanego w kroku 4

```
Receive-Job -Id {id}
```

- h. Pobierz wynik wszystkich jobów

```
Get-job | Receive-job
```

- i. Usuń wszystkie joby

```
Get-Job | Remove-Job
```

- j. Przygotuj trigger do uruchomienia zadania zaplanowanego o godzinie 15:00 codziennie

```
$trigger = New-ScheduledTaskTrigger -Daily -At 15:00  
Register-ScheduledJob -Name "job" -Trigger $trigger -ScriptBlock
```



```
{write-host "job"}
```

- k. Uruchom zaplanowane zadanie

```
Start-ScheduledTask -TaskName "job"
```

- l. Oczekuj na zakończenie zadania (np. 10 minut)

```
Start-Sleep -Seconds 600
```

- m. Wyświetl listę zdefiniowanych zadań

```
Get-ScheduledTask
```

- n. Wyrejestruj zadanie

```
Unregister-ScheduledTask -TaskName {nazwa}
```

## Zakończenie

Wewnątrz maszyny wirtualnej uruchom **Stop-Transcript**

Po wykonywaniu zadania w folderze Imie\_nazwisko\_ucznia zapisz pliki wynikowe dokumentujące wykonane zadania.

A. Wewnątrz maszyny wirtualnej otwórz plik **C:\imienazwisko\_ppk1\_grnr.txt** zaznacz jego zawartość i wybierz kopiuj, na pulpicie maszyny fizycznej w utworzonym folderze Imie\_nazwisko\_ucznia\_pk utwórz plik **imienazwisko\_ppk1\_grnr.txt** i wybierz wklej. Zamknij pliki.

B. Wewnątrz maszyny wirtualnej otwórz plik **C:\imienazwisko\_rpk1\_grnr.txt** w którym umieściłeś odpowiedzi do zadań zaznacz jego zawartość i wybierz kopiuj, na pulpicie maszyny fizycznej w utworzonym folderze Imie\_nazwisko\_ucznia\_pk utwórz plik **imienazwisko\_rpk1\_grnr.txt** i wybierz wklej. Upewnij się, że znajdują się tam odpowiedzi do zadań. Zamknij pliki.

## Uwaga:

Na pulpicie maszyny fizycznej znajduje się folder o nazwie **Imie\_nazwisko\_ucznia\_pk**. Aby go spakować, możesz skorzystać z wbudowanego w system Windows narzędzia Zip, wybierając opcję **'Wyślij do'** i następnie po prawokliku na tym folderze **'Folder skompresowany (zip)'**.

Do sprawdzenia oddajemy spakowany folder Imie\_nazwisko\_ucznia na pulpicie maszyny fizycznej w którym znajdują się dwa pliki:

**imienazwisko\_ppk1\_grnr.txt**

**imienazwisko\_rpk1\_grnr.txt**

<https://isobczak.zsl.gda.pl/powershell/11%20Powershell%20remoting/37/Sesje%20-%20KLUCZ.pdf>