Komendy CLI

- Sprawdzenie samej listy IPv4 kart sieciowych
- Reguły w firewallu dla ECHO / PING / ICMP
- Czyszczenie kosza wszystkich userów
- Sprawdzenie, kto wykonał restart OS
- Czyszczenie starych komponentów WinSxS
- Manualne wymuszenie synchronizacji AD on premis z EntraID
- Podłączanie się do WinRM - Powershell
- Instalacja Telnet poprzez powershell
- Zaszyfrowanie hasła dla secure-string w Powershell
- Zdalne uruchamianie skryptów PS
- Zarządzanie RemoteApps w Remote Desktop Services (RDS)
- Rebuild performance counter oraz resynchronizacja z WMI
- Rozszerzenie StorageTier (wymagane do powiększenia wolumenów przy S2D)
- Zmiana nazwy dla ClusterSharedVolume
- Wgranie danych poprzez WinRM (Powershell Remoting)
- Instalacja MSI w quiet

Sprawdzenie samej listy IPv4 kart sieciowych

Wszystkie komendy uruchamiamy w powershell.

Komenda ipconfig:

```
ipconfig /all | findstr /I "Adapter | Descripton | IPv4"
```

Komenda Get-NetAdapter, z ładniejszym formatowaniem)

```
Get-NetAdapter | ForEach-Object {
    $adapterName = $_.InterfaceAlias
    $ipv4 = (Get-NetIPAddress -InterfaceAlias $adapterName -AddressFamily IPv4).IPAddress
    [PSCustomObject]@{
        "Karta sieciowa" = $adapterName
        "Adres IPv4"     = $ipv4 -join ", "
    }
} | Format-Table -AutoSize
```

Reguły w firewallu dla ECHO / PING / ICMP

Dodanie reguł

```
New-NetFirewallRule -DisplayName "PING / ECHO / ICMP v4" -Direction Inbound -Protocol ICMPv4 -IcmpType 8 -Action Allow
New-NetFirewallRule -DisplayName "PING / ECHO / ICMP v6" -Direction Inbound -Protocol ICMPv6 -IcmpType 8 -Action Allow
```

Usunięcie reguł:

```
Remove-NetFirewallRule -DisplayName "PING / ECHO / ICMP v4"
Remove-NetFirewallRule -DisplayName "PING / ECHO / ICMP v6"
```

Sprawdzenie czy reguły o danej nazwie występują:

```
Get-NetFirewallRule | Where-Object DisplayName -like "*PING*"
```

Czyszczenie kosza wszystkich userów

Komendę uruchamiamy jako Administrator w cmd / powershell.

```
rd /s /q C:/$Recycle.bin
```

- /s oznacza usunięcie podfolderów
- /q oznacza quiet mode - nie będzie pytał o potwierdzenie

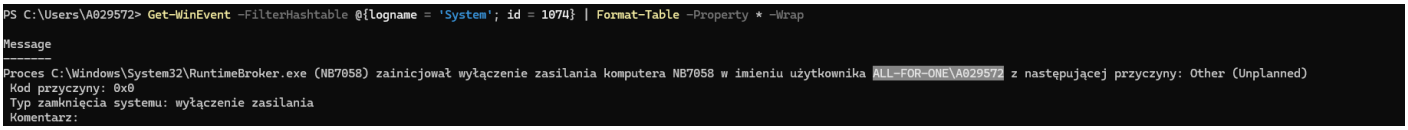
Docelowy dysk z C:/ można zmienić na odpowiedni

Sprawdzenie, kto wykonał restart OS

Komendę uruchamiamy jako Administrator w powershell.

```
Get-WinEvent -FilterHashtable @{logname = 'System'; id = 1074} | Format-Table -Property * -Wrap
```

Przykładowy wynik:



Czyszczenie starych komponentów WinSxS

Komendę uruchamiamy jako Administrator w cmd / powershell.

```
Dism.exe /online /Cleanup-Image /StartComponentCleanup /ResetBase
```

Manualne wymuszenie synchronizacji AD on premis z EntraID

Komendę uruchamiamy jako Administrator w powershell na kontrolerze domeny ze skonfigurowanymi odpowiednimi połączeniami do Entra ID.

```
Start-ADSyncSyncCycle -PolicyType Delta
```

Podłączanie się do WinRM - Powershell

> Uruchamiamy z poziomu powershell, po uruchomieniu usługi WinRM

```
Enter-PSSession -ComputerName "NazwaKomputera" -Credential (Get-Credential)
```

Instalacja Telnet poprzez powershell

Komendę uruchamiamy jako Administrator w powershell.

```
pkgmgr /iu:"TelnetClient"
```

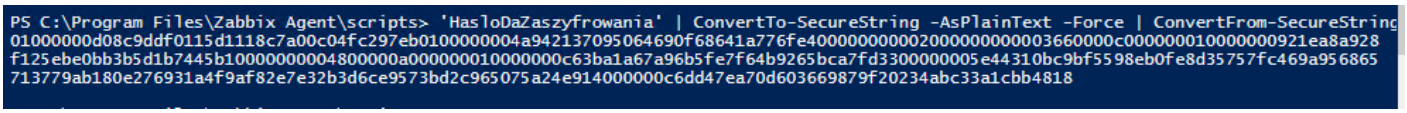
Zaszyfrowanie hasła dla secure-string w Powershell

Komendę uruchamiamy jako użytkownik który będzie z tego hasła korzystał. Hasło wygenerowane będzie działać poprawnie tylko i wyłącznie użytkownikiem z którego wykonamy komendę.

Jeżeli powershell uruchomimy **jako Administrator**, to kontem wykonawczym np. w harmonogramie zadań **musi być administrator.**

```
'HasloDoZaszyfrowania' | ConvertTo-SecureString -AsPlainText -Force | ConvertFrom-SecureString
```

Przykładowy wynik:



Zdalne uruchamianie skryptów PS

Wykonujemy z poziomu Powershell jako użytkownik z uprawnieniami administracyjnymi na komputerze docelowym:

```
Invoke-Command -ComputerName <computer_name> -scriptblock {PowerShell.exe -ExecutionPolicy Bypass -File "/local/path/to/script.ps1"}
```

Zarządzanie RemoteApps w Remote Desktop Services (RDS)

RemoteApp to funkcja Remote Desktop Services (RDS), która pozwala na publikowanie aplikacji zdalnych, tak aby wyglądały i działały jak lokalne programy na komputerze użytkownika. Dzięki temu użytkownicy mogą uruchamiać aplikacje z serwera RDS bez potrzeby pełnego dostępu do pulpitu zdalnego.

Komendy uruchamiamy z poziomu powershell, z zainstalowanym modułem RemoteDesktop, najlepiej od razu na serwerze z zainstalowaną rolą Remote Desktop Services

Wylistowanie wszystkich aplikacji w kolekcji

```
# Wskazanie nazwy kolekcji z której pobieramy aplikacje
$collectionName = "Nazwa Kolekcji"

# Zaimportowanie modułu RemoteDesktop
Import-Module RemoteDesktop

# Pobieranie wszystkich RemoteApp z kolekcji
$remoteApps = Get-RDRemoteApp -CollectionName $collectionName

# Wyświetlanie pobranych aplikacji RemoteApp
$remoteApps | Select-Object Alias, DisplayName, FilePath
```

Dodanie nowej aplikacji do kolekcji

```
# Pobieranie nazwy kolekcji do której chcemy dodać aplikację
$collectionName = "Nazwa Kolekcji"

# Pobieranie parametrów nowej aplikacji
$lnkFilePath = "C:\Ścieżka\do\aplikacji\Skrót.lnk"
$iconFilePath = "C:\Ścieżka\do\pliku\ikony.ico"
```

```
$appAlias = "Alias aplikacji w RDS"
$appDisplayName = "Nazwa aplikacji w RDS wyświetlana dla użytkowników"

# Importowanie potrzebnego modułu powershell
Import-Module RemoteDesktop

# Dodanie nowej aplikacji
New-RDRemoteApp -CollectionName $collectionName -Alias $appAlias -DisplayName $appDisplayName -FilePath
$lnkFilePath -IconPath $iconFilePath
```

Zmiana nazwy aplikacji w kolekcji

```
# Pobieranie nazwy kolekcji i aplikacji do zmiany
$collectionName = "Nazwa Kolekcji"
$alias = "Stara nazwa aplikacji"

# Pobieranie parametrów do zmiany
$newDisplayName = "Nowa nazwa aplikacji"
$newProgramPath = "C:\Ścieżka\do\aplikacji.lnk" #Bądź do skrótu

# Importowanie odpowieniego modułu powershell
Import-Module RemoteDesktop

# Zmiana nazwy aplikacji
Set-RDRemoteApp -CollectionName $collectionName -Alias $alias -DisplayName $newDisplayName -FilePath
$newProgramPath
```

Usunięcie aplikacji z kolekcji

```
# Pobieranie nazwy kolekcji i aplikacji do usunięcia
$collectionName = "Nazwa Kolekcji"
$alias = "Aplikacja do usunięcia"

# Importowanie odpowieniego modułu powershell
Import-Module RemoteDesktop

# Usunięcie aplikacji z kolekcji
Remove-RDRemoteApp -CollectionName $collectionName -Alias $alias
```

Rebuild performance counter oraz resynchronizacja z WMI

> Komendy uruchamiamy w CMD jako administrator

```
cd C:\Windows\system32
lodctr /R

cd C:\Windows\SysWOW64
lodctr /R
```

Resynchronizacja performance counterów z WMI

```
WINMGMT.EXE /RESYNCPERF
```

Rozszerzenie StorageTier (wymagane do powiększenia wolumenów przy S2D)

> Na podstawie: windowsserverdocs/WindowsServerDocs/storage/storage-spaces/manage-volumes.md at main · MicrosoftDocs/windowsserverdocs · GitHub

Powiększanie StoregeTier:

Pobieranie wszystkich VirtualDisks:

```
Get-VirtualDisk
```

```
PS C:\Users\admin_tomkon> Get-VirtualDisk

FriendlyName              ResiliencySettingName FaultDomainRedundancy OperationalStatus HealthStatus  Size FootprintOnPool StorageEfficiency
------------              --------------------- -------------------- ----------------- ------------  ---- --------------- -----------------
ClusterPerformanceHistory Mirror                1                    OK                Healthy      32 GB           65 GB            49,23%
S2D LocalDisks                                                       OK                Healthy      10 TB        20.01 TB            49,98%
```

Pobieranie nazwy StorageTier

```
Get-VirtualDisk "NazwaDyskuJeżeliZnamy" | Get-StorageTier | Select FriendlyName
```

```
PS C:\Users\admin_tomkon> Get-VirtualDisk | Get-StorageTier | Select FriendlyName

FriendlyName
------------
S2D LocalDisks-Capacity
```

Sprawdzenie parametrów konkretnego StorageTier

```
Get-StorageTier -FriendlyName "NazwaStorageTier"
```

Rozszerzanie odpowiedniego StorageTier do dedykowanej wielkości (MB/GB/TB itd)

```
Get-StorageTier -FriendlyName "NazwaStorageTier" | Resize-StorageTier -Size (10TB)
```

```
PS C:\Users\admin_tomkon> Get-StorageTier -FriendlyName "S2D LocalDisks-Capacity"

FriendlyName            TierClass MediaType ResiliencySettingName FaultDomainRedundancy   Size FootprintOnPool StorageEfficiency
------------            --------- --------- --------------------- --------------------   ---- -------------- -----------------
S2D LocalDisks-Capacity Capacity  SSD       Mirror                1                    4.5 TB         9.01 TB            49,96%


PS C:\Users\admin_tomkon> Get-StorageTier -FriendlyName "S2D LocalDisks-Capacity" | Resize-StorageTier -Size (10TB)
PS C:\Users\admin_tomkon> Get-StorageTier -FriendlyName "S2D LocalDisks-Capacity"

FriendlyName            TierClass MediaType ResiliencySettingName FaultDomainRedundancy  Size FootprintOnPool StorageEfficiency
------------            --------- --------- --------------------- --------------------  ---- -------------- -----------------
S2D LocalDisks-Capacity Capacity  SSD       Mirror                1                    10 TB          20 TB            50,00%
```

Powiększenie wolumenu :

Sprawdzenie nazwy dysku wirtualnego:

```
Get-VirtualDisk
```

```
PS C:\Users\admin_tomkon> Get-VirtualDisk

FriendlyName             ResiliencySettingName FaultDomainRedundancy OperationalStatus HealthStatus  Size FootprintOnPool StorageEfficiency
------------             --------------------- -------------------- ----------------- ------------  ---- -------------- -----------------
ClusterPerformanceHistory Mirror               1                    OK                Healthy      32 GB          65 GB            49,23%
S2D LocalDisks                                                      OK                Healthy      20 TB          40 TB            50,00%
```

Samo powiększanie wolumenu:

```
$VirtualDisk = Get-VirtualDisk "NazwaDysku"
$Partition = $VirtualDisk | Get-Disk | Get-Partition | Where PartitionNumber -Eq 2
$Partition | Resize-Partition -Size ($Partition | Get-PartitionSupportedSize).SizeMax
```

```
PS C:\Users\admin_tomkon> $VirtualDisk = Get-VirtualDisk "S2D LocalDisks"
PS C:\Users\admin_tomkon> $Partition = $VirtualDisk | Get-Disk | Get-Partition | Where PartitionNumber -Eq 2
PS C:\Users\admin_tomkon> $Partition | Resize-Partition -Size ($Partition | Get-PartitionSupportedSize).SizeMax
```

Wynik Get-VirtualDisk po powiększeniu:

```
PS C:\Users\admin_tomkon> Get-VirtualDisk

FriendlyName             ResiliencySettingName FaultDomainRedundancy OperationalStatus HealthStatus  Size FootprintOnPool StorageEfficiency
------------             --------------------- -------------------- ----------------- ------------  ---- -------------- -----------------
ClusterPerformanceHistory Mirror               1                    OK                Healthy      32 GB          65 GB            49,23%
S2D LocalDisks                                                      OK                Healthy      20 TB          40 TB            50,00%
```

Zmiana nazwy dla ClusterSharedVolume

> Polecenia wykonujemy na hoście Hyper-V, z poziomu powershell, uruchomionego z uprawnieniami administratora

Pobieranie nazwy do zmiany:

```
Get-ClusterSharedVolume
```

```
PS C:\Users\admin_tomkon> Get-ClusterSharedVolume

Name                                State              Node
----                                -----              ----
Cluster Disk 1                      Online(Redirected) FT-S002
Cluster Virtual Disk (S2D LocalDisks) Online           FT-S001
```

```
(Get-ClusterSharedVolume -Name "NazwaDoZmiany").Name = "NowaNazwa"
```

```
PS C:\Users\admin_tomkon> (Get-ClusterSharedVolume -Name "Cluster Disk 1").Name = "iSCSI QNAP DISK"
PS C:\Users\admin_tomkon> Get-ClusterSharedVolume

Name                                State              Node
----                                -----              ----
Cluster Virtual Disk (S2D LocalDisks) Online           FT-S001
iSCSI QNAP DISK                     Online(Redirected) FT-S002
```

Wgranie danych poprzez WinRM (Powershell Remoting)

> Wcześniej musi być uruchomiona usługa WinRM, oraz musimy mieć włączony dostęp sieciowy do danego komputera

```
$cred = Get-Credential
$session = New-PSSession -ComputerName Nazwa-Komputera -Credential $cred

Copy-Item  "C:\Temp\Plik Źródłowy.plik" `
    -Destination "C:\Temp\Plik docelowy.plik" `
    -ToSession $session
```

Instalacja MSI w quiet

> Uruchamiamy z poziomu CMD bądź Powershell z uprawnieniami administraotra

```
msiexec /i "C:\Windows\Temp\Plik.msi" /qn /norestart /L*v "C:\Windows\Temp\Plik.log"
```