Skonfigurowanie zdalnego czyszczenia na urządzeniach

Przygotowanie

Enable-PSRemoting -Force -SkipNetworkProfileCheck

New-Item -Path 'C:\ProgramData\CustomScripts' -ItemType Directory -Force

A. Wyczyść urządzenie (skrypt) bez usługi Intune

Wykonaj notatkę, zaprezentuj i omów wykonane czynności.

Wyczyszczenie/zresetowanie urządzenia z systemem Windows 10 w skryptowy sposób, gdy nie korzystasz z usługi Intune. Sposób PowerShell

Skrypt PowerShell składa się z 3 części

1. Zresetuj część WMI urządzenia
2. Eksport autopilota i część e-mail (lub automatyczne przesyłanie)
3. Pobieranie narzędzi Sysinternal i ich rozpakowywanie

Uruchom skrypt… tworzysz 2 skrypty. Uruchamiasz pierwszy skrypt jako admin (monit UAC), a po pierwszym skrypcie uruchamiasz drugi skrypt (Reset Device) jako system.

1. Wykonaj zdalne resetowanie urządzenia za pomocą Windows Management Instrumentation (WMI).

#MDM WMI Bridge part

$reset =

@'

$namespaceName = "root\cimv2\mdm\dmmap"

$className = "MDM_RemoteWipe"

$methodName = "doWipeMethod"

$session = New-CimSession

$params = New-Object Microsoft.Management.Infrastructure.CimMethodParametersCollection

$param = [Microsoft.Management.Infrastructure.CimMethodParameter]::Create("param", "", "String", "In")

$params.Add($param)

$instance = Get-CimInstance -Namespace $namespaceName -ClassName $className -Filter “ParentID=’./Vendor/MSFT’ and InstanceID=’RemoteWipe”’

$session.InvokeMethod($namespaceName, $instance, $methodName, $params)

‘@

To jest fragment skryptu PowerShell, który wykonuje pewne operacje związane z Mostkiem MDM WMI (MDM WMI Bridge), czyli narzędziem pozwalającym na zarządzanie urządzeniami z systemem Windows 10 i 11 za pomocą rozwiązań MDM (Mobile Device Management). Skrypt wykonuje następujące operacje:

Tworzy nową zmienną $reset, która zawiera wielowierszowy ciąg znaków (@' i '@ na końcu), w którym znajduje się kod PowerShell.

Ustawia wartość zmiennej $namespaceName na "root\cimv2\mdm\dmmap", co oznacza, że skrypt będzie działał w przestrzeni nazw WMI dla MDM.

Ustawia wartość zmiennej $className na "MDM_RemoteWipe", co oznacza, że skrypt będzie korzystał z klasy MDM_RemoteWipe, która pozwala na zdalne usuwanie danych z urządzenia.

Ustawia wartość zmiennej $methodName na "doWipeMethod", co oznacza, że skrypt będzie wywoływał metodę doWipeMethod, która służy do wykonywania zdalnego czyszczenia urządzenia.

Tworzy nową sesję CIM (Common Information Model) za pomocą polecenia New-CimSession, która umożliwia nawiązanie połączenia z MDM WMI Bridge.

Tworzy kolekcję parametrów dla metody za pomocą polecenia New-Object Microsoft.Management.Infrastructure.CimMethodParametersCollection, która służy do przechowywania parametrów.

Tworzy nowy parametr za pomocą polecenia [Microsoft.Management.Infrastructure.CimMethodParameter]::Create i dodaje go do kolekcji parametrów.

Pobiera instancję klasy MDM_RemoteWipe, która ma wartość parametru InstanceID równą "RemoteWipe" i ParentID równą "./Vendor/MSFT".

Wywołuje metodę doWipeMethod z parametrami za pomocą polecenia $session.InvokeMethod, które wykonuje zdalne wywołanie metody na urządzeniu (opcjonalne parametry) na tej instancji, co skutkuje zresetowaniem urządzenia.

Zdalne resetowanie jest procesem przywracania urządzenia do stanu fabrycznego, co może być użyteczne w przypadku utraty urządzenia lub gdy urządzenie zostanie skompromitowane. Jest to często stosowana funkcja w zarządzaniu urządzeniami korporacyjnymi, pozwalająca na ochronę danych poprzez ich usunięcie zdalnie.

2. Eksportuje autopilota i wyśle wiadomość do pliku CSV, "autopilot" odnosi się do **Windows Autopilot**, usługi firmy Microsoft, która umożliwia uproszczone wdrażanie i konfigurację nowych urządzeń z systemem Windows 10 i 11, bez konieczności fizycznego dotykania urządzenia.

```
$start =

@'

$ProgressPreference = "SilentlyContinue"

$ErrorActionPreference= 'silentlycontinue'

$OriginalPref = $ProgressPreference

New-Item -Path c:\programdata\customscripts -ItemType Directory -Force -Confirm:$false | out-null

install-packageprovider -name nuget -minimumversion 2.8.5.201 -force | out-null

Save-Script -Name Get-WindowsAutoPilotInfo -Path c:\ProgramData\CustomScripts -force | out-null

Autopilot Info Uploaden

Upload-WindowsAutopilotDeviceInfo -TenantName $tenantname -OrderIdentifier "AADUserDriven" -Verbose

start-sleep -s 10

c:\ProgramData\CustomScripts\Get-WindowsAutoPilotInfo.ps1 -OutputFile c:\ProgramData\CustomScripts\MyComputer.csv

$PCName = $env:COMPUTERNAME

$EmailBody = $event
```

```
$EmailFrom = “$PCName@domeinnaam.pl”

$EmailTo = “autopilot@domeinnaam.pl”

$EmailSubject = “Autopilot CSV”

$SMTPServer = “tennantname.onmicrosoft.com”

Write-host “Sending Email”

Send-MailMessage -From $EmailFrom -To $EmailTo -Subject $EmailSubject -SmtpServer $SMTPServer -Attachments c:\ProgramData\CustomScripts\mycomputer.csv

#Start PowerShell session as system

Start-Process -FilePath “c:\ProgramData\CustomScripts\pstools\psexec.exe” -windowstyle hidden -ArgumentList ‘-i -s cmd /c “powershell.exe -ExecutionPolicy Bypass -file c:\programdata\customscripts\reset.ps1″‘

‘@
```

Polecenie to jest skryptem PowerShell, który wykonuje kilka czynności związanych z pobraniem informacji o urządzeniu Windows i przesłaniem tych informacji na serwer.

Linie kodu można opisać następująco:

Ustawia zmienną $ProgressPreference na "SilentlyContinue", aby zignorować wszelkie komunikaty postępu wyświetlane podczas uruchamiania skryptu.

Ustawia zmienną $ErrorActionPreference na "silentlycontinue", aby zignorować wszelkie błędy, które mogą wystąpić podczas wykonywania skryptu.

Zapisuje obecne ustawienia zmiennej $ProgressPreference do $OriginalPref.

Tworzy nowy katalog "c:\programdata\customscripts", jeśli nie istnieje, z opcją potwierdzenia użytkownika.

Instaluje dostawcę pakietów NuGet o nazwie "nuget" o minimalnej wersji 2.8.5.201, wymuszając zignorowanie błędów.

Pobiera skrypt i zapisuje skrypt o nazwie "Get-WindowsAutoPilotInfo" w katalogu "c:\ProgramData\CustomScripts", wymuszając zignorowanie błędów. Ten skrypt jest używany do zbierania informacji o urządzeniu, które są potrzebne do Windows Autopilot.

Pobiera informacje o urządzeniu do usługi Windows Autopilot w chmurze Microsoft Azure i zapisuje je w pliku CSV o nazwie "MyComputer.csv" w katalogu "c:\ProgramData\CustomScripts".

Uruchamia skrypt Get-WindowsAutoPilotInfo.ps1, który generuje plik CSV z informacjami o urządzeniu.

Zapisuje nazwę komputera w zmiennej $PCName.

Tworzy zmienną $EmailBody i przypisuje jej wartość $event.

Tworzy zmienną $EmailFrom i przypisuje jej wartość "$PCName@domeinnaam.pl".

Tworzy zmienną $EmailTo i przypisuje jej wartość "autopilot@domeinnaam.pl".

Tworzy zmienną $EmailSubject i przypisuje jej wartość "Autopilot CSV".

Ustawia nazwę serwera SMTP na "tennantname.onmicrosoft.com".

Wyświetla napis "Sending Email".

Wysyła wiadomość e-mail zawierającą plik CSV "mycomputer.csv" zawierającym informacje o urządzeniu do adresu e-mail skonfigurowanego adresata "autopilot@domeinnaam.pl".

Uruchamia PowerShell z uprawnieniami systemowymi z wykorzystaniem narzędzia psexec.exe z PSTools, a następnie wykonuje znajdujący się w katalogu "c:\programdata\customscripts" skrypt reset.ps1 z uprawnieniami systemowymi.

3. Pobieranie Sysinternals i uruchamianie skryptu jako system

Ten skrypt PowerShell odpowiada za eksportowanie dwóch skryptów (reset.ps1 i start.ps1) do folderu ProgramData na lokalnym komputerze, pobieranie i rozpakowywanie narzędzi Sysinternals Suite, akceptowanie licencji EULA dla narzędzia Psexec i uruchomienie skryptu startowego jako administrator.

Poniżej znajduje się wyjaśnienie poszczególnych poleceń:

1. Export skryptów do folderu ProgramData:

#Export script to programdata folder

Utwórz folder CustomScripts, jeśli nie istnieje

```
$folderPath = Join-Path $env:ProgramData "CustomScripts"

if (-not (Test-Path $folderPath)) {

    New-Item -Path $folderPath -ItemType Directory

}

# Zapisz skrypt reset.ps1 do folderu CustomScripts

$filePath = Join-Path $folderPath "reset.ps1"

Out-File -FilePath $filePath -Encoding unicode -Force -InputObject $reset -Confirm:$false

Out-File -FilePath $(Join-Path $env:ProgramData CustomScripts\start.ps1) -Encoding unicode -Force -InputObject $start -Confirm:$false
```

Polecenie to korzysta z cmdletu Out-File w celu zapisania dwóch skryptów (reset.ps1 i start.ps1) do folderu ProgramData\CustomScripts na lokalnym komputerze. Zmienna $env:ProgramData zawiera ścieżkę do folderu ProgramData. Opcja -Encoding określa kodowanie pliku, a opcja -Force powoduje nadpisanie istniejącego pliku. Opcja -InputObject określa, co zostanie zapisane, a opcja -Confirm:$false powoduje pominięcie pytania o potwierdzenie zapisu.

2. Akceptacja licencji EULA dla narzędzia Psexec:

```
#Accepteula Psexec

reg.exe ADD HKCU\Software\Sysinternals /v EulaAccepted /t REG_DWORD /d 1 /f | out-null
```

Polecenie to dodaje wartość klucza rejestru EulaAccepted do gałęzi HKCU\Software\Sysinternals, która akceptuje licencję EULA dla narzędzia Psexec. Opcja /t określa typ danych klucza rejestru, /d określa wartość danych, a /f powoduje wymuszenie zapisu.

3. Pobranie i rozpakowanie narzędzi Sysinternals Suite:

```
#Sysinternals download part

invoke-webrequest -uri: “https://download.sysinternals.com/files/SysinternalsSuite.zip”  -outfile “c:\programdata\customscripts\pstools.zip”  | out-null

Expand-Archive c:\programdata\customscripts\pstools.zip  -DestinationPath c:\programdata\customscripts\pstools  -force | out-null
```

Polecenie to używa cmdletu Invoke-WebRequest w celu pobrania pliku ZIP zawierającego narzędzia Sysinternals Suite i zapisuje go jako plik pstools.zip w folderze ProgramData\CustomScripts na lokalnym komputerze. Opcja -uri określa adres URL pliku ZIP, a -outfile określa lokalizację i nazwę pliku, który zostanie zapisany. Polecenie następnie używa cmdletu Expand-Archive do rozpakowania pliku ZIP w folderze ProgramData\CustomScripts\pstools. Opcja -DestinationPath określa ścieżkę docelową, a -force wymusza nadpisanie istniejących plików.

4. Uruchomienie skryptu startowego jako administrator

#Start Powershell Script as Admin

Start-Process powershell -ArgumentList '-noprofile -file c:\programdata\customscripts\start.ps1' -verb RunAs

Polecenie to uruchamia skrypt Powershell jako administrator.

Start-Process jest cmdlet'em w PowerShell, który pozwala na uruchamianie nowych procesów. W tym przypadku, Start-Process zostaje użyte do uruchomienia skryptu Powershell.

-ArgumentList określa listę argumentów, które będą przekazywane do Powershell. W tym przypadku -noprofile -file c:\programdata\customscripts\start.ps1 jest przekazywane jako argumenty do Powershell. -noprofile wyłącza załadowanie domyślnego profilu użytkownika, co oznacza, że skrypt nie będzie miał dostępu do żadnych zmiennych środowiskowych, które są zazwyczaj dostępne dla użytkownika. -file c:\programdata\customscripts\start.ps1 określa ścieżkę do pliku skryptu, który zostanie uruchomiony.

-verb RunAs jest opcją, która informuje PowerShell, że powinien uruchomić nowy proces z uprawnieniami administratora. Opcja ta wyświetla okno dialogowe UAC (User Account Control), które wymaga potwierdzenia uprawnień administratora przed kontynuowaniem.

Podsumowując, polecenie to uruchamia skrypt Powershell jako administrator, wyłączając przy tym załadowanie domyślnego profilu użytkownika i wymagając potwierdzenia uprawnień administratora przed uruchomieniem procesu.

Z poniższej części wykonaj tylko notatkę

B. Wyczyść urządzenie mobilne za pomocą Powershell

Zdalnie wyczyść urządzenie mobilne

Jeśli urządzenie zostanie zgubione lub skradzione, możesz usunąć poufne dane organizacji i zapobiec dostępowi do zasobów organizacji platformy Microsoft 365, wymazując dane z portalu zgodności Microsoft Purview > Zapobieganie utracie danych > Zarządzanie urządzeniami. Możesz wykonać czyszczenie selektywne, aby usunąć tylko dane organizacyjne lub pełne czyszczenie, aby usunąć wszystkie informacje z urządzenia i przywrócić je do ustawień fabrycznych.

Gdy musisz zdalnie wyczyścić urządzenie przenośne połączone z usługą Exchange Online, możesz szybko wykonać zadanie za pomocą programu Powershell. Za pomocą polecenia cmdlet Clear-Mobile usuniesz wszystkie dane z urządzenia, co spowoduje również usunięcie wszelkich znajdujących się na nim zdjęć, aplikacji lub innych danych osobowych.

Aby to zrobić, musimy połączyć się z Exchange Online, używamy do tego skryptu łącznika.

1. Lista wszystkich urządzeń

Musimy uzyskać identyfikator urządzenia. Aby wyświetlić listę wszystkich urządzeń użytkownika, uruchom następujące polecenie cmdlet. Dzięki temu uzyskasz tożsamość, której będziemy potrzebować później, oraz dodatkowe informacje, dzięki którym będziesz mógł wybrać właściwe urządzenie.

```
#Connect to Exchange Online with the connector script

ConnectTo-ExchangeOnline.ps1

#List all connected devices

Get-MobileDevice -Mailbox <emailaddress> | select Identity, FriendlyName, DeviceOS, DeviceType | FT
```

Pierwsze polecenie "ConnectTo-ExchangeOnline.ps1" nawiązuje połączenie z usługą Exchange Online, która jest częścią platformy Microsoft Office 365. W tym celu wykorzystuje skrypt o nazwie "ConnectTo-ExchangeOnline.ps1". Jest to prawdopodobnie skrypt PowerShell, który zawiera zestaw poleceń i konfiguracji umożliwiających nawiązanie połączenia z Exchange Online.

Drugie polecenie "Get-MobileDevice -Mailbox <emailaddress> | select Identity, FriendlyName, DeviceOS, DeviceType | FT" pozwala na wylistowanie wszystkich urządzeń mobilnych połączonych z określonym skrzynką pocztową użytkownika, którego adres email jest podany w miejsce "<emailaddress>".

Polecenie "Get-MobileDevice" pobiera informacje o urządzeniach mobilnych połączonych z określonym kontem pocztowym, a polecenie "| select Identity, FriendlyName, DeviceOS, DeviceType" filtruje wyniki

i wybiera tylko te informacje, które są potrzebne (czyli identyfikator urządzenia, nazwa, system operacyjny i typ urządzenia).

Na końcu "| FT" wykorzystuje polecenie "Format-Table", aby wyniki zostały sformatowane w postaci tabeli, co ułatwia odczytanie wyników.

2. Wyczyść dane urządzenia

Aby faktycznie usunąć dane z urządzenia, skopiuj tożsamość i uruchom następujące polecenie cmdlet

Clear-MobileDevice -Identity '<identity>' -NotificationEmailAddresses <emailaddress>

Opcja notificationEmailAddress umożliwia otrzymanie potwierdzenia po zakończeniu czyszczenia. Możesz podać wiele adresów oddzielonych przecinkami.

Polecenie "Clear-MobileDevice" jest komendą w PowerShell, która służy do zdalnego wyczyszczenia zawartości urządzenia mobilnego.

Parametr "-Identity" określa, które urządzenie ma być wyczyszczone, a jego wartość to <identity>. Można to zrobić podając ID urządzenia lub nazwę użytkownika powiązaną z urządzeniem.

Parametr "-NotificationEmailAddresses" określa adresy e-mail, na które mają zostać wysłane powiadomienia po wykonaniu czyszczenia. Wartość tego parametru to <emailaddress>, a można podać jedno lub wiele adresów e-mail oddzielonych przecinkami.

W skrócie, polecenie "Clear-MobileDevice" usuwa zawartość urządzenia mobilnego, a "-Identity" określa urządzenie, które ma być wyczyszczone, a "-NotificationEmailAddresses" określa adresy e-mail, na które mają zostać wysłane powiadomienia po wykonaniu czyszczenia.