PowerShell & PSADT

Date: 09-August-2025

Batch: WiproNGA_DWS_B5_25VID2550

Candidate Name: Sooraj B

User ID: **34753**

1. PSADT (PowerShell App Deployment Toolkit) Overview

The PowerShell App Deployment Toolkit simplifies software deployments in enterprise environments by offering a structured and flexible scripting framework. It's widely used with SCCM and Intune.

Key Components:

- **Files** → Application installers
- **SupportFiles** → Dependencies/configs
- **Deploy-Application.ps1** → Main script for install, uninstall, and updates
- **Logging** → Stored in C:\Windows\Logs\Software (viewable in CMTrace)

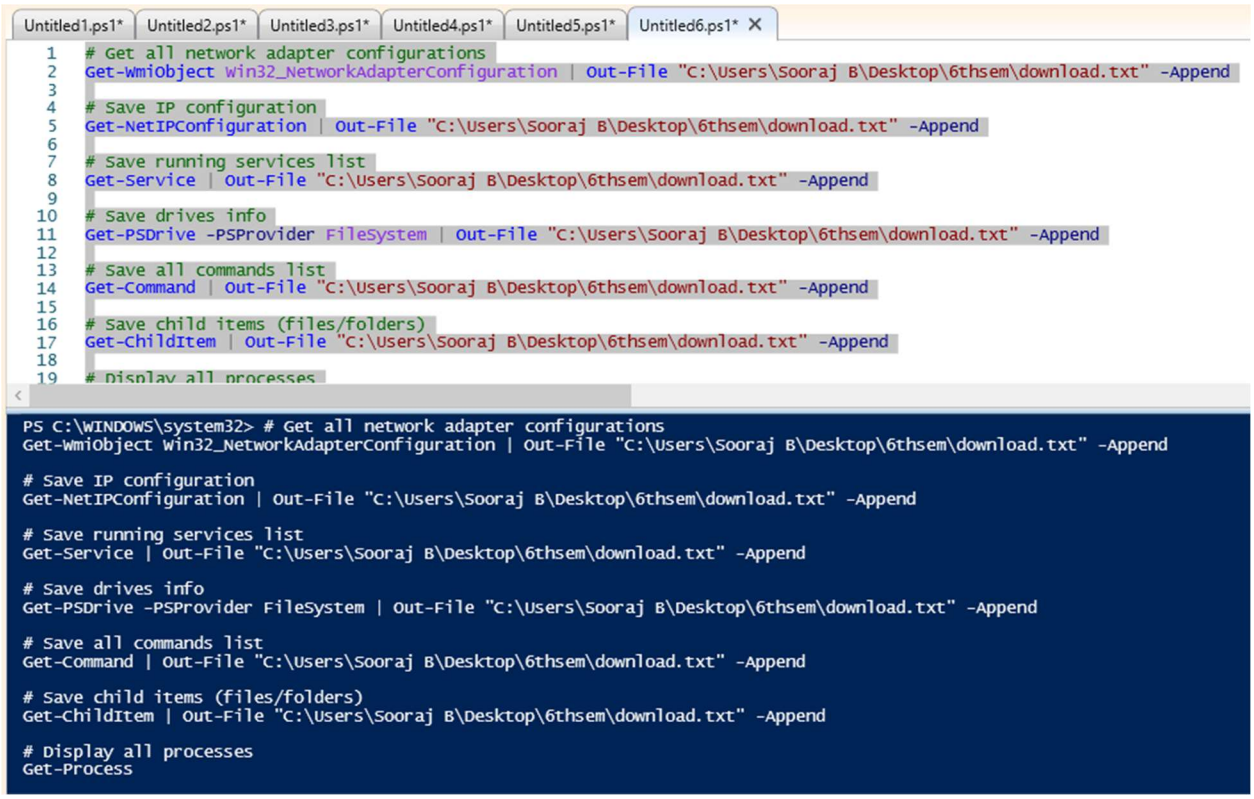
Example Install Code:

powershell

CopyEdit

Execute-MSI -Action Install -Path "AppInstaller.msi"

2. PowerShell Scripting Constructs



PowerShell scripting constructs are the core of automation.

Example Variables and Arrays:

powershell

CopyEdit

$name = "John" $age = 30 $fruits = "apple","banana" $fruits += "kiwi"

Example Condition:

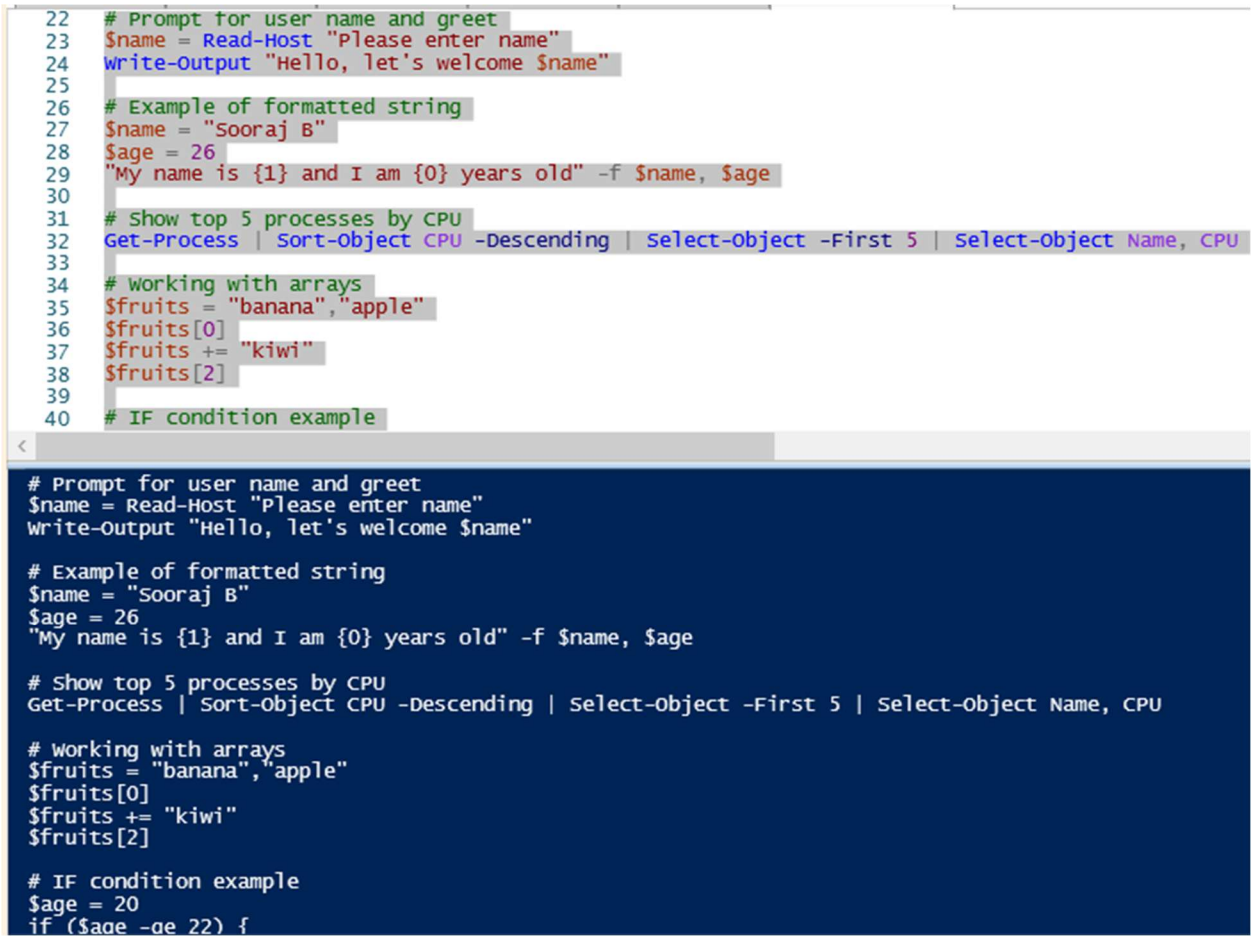
powershell

CopyEdit

if ($age -ge 18) { Write-Host "Adult" } else { Write-Host "Minor" }

3. WMI & CIM

Used to query system hardware and OS info.



```powershell
# Prompt for user name and greet
$name = Read-Host "Please enter name"
Write-Output "Hello, let's welcome $name"

# Example of formatted string
$name = "Sooraj B"
$age = 26
"My name is {1} and I am {0} years old" -f $name, $age

# Show top 5 processes by CPU
Get-Process | Sort-Object CPU -Descending | Select-Object -First 5 | Select-Object Name, CPU

# Working with arrays
$fruits = "banana","apple"
$fruits[0]
$fruits += "kiwi"
$fruits[2]

# IF condition example
$age = 20
if ($age -ge 22) {
```

Examples:

powershell

CopyEdit

Get-CimInstance Win32_OperatingSystem | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append Get-CimInstance Win32_Processor | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append

4. Modularization in PowerShell

Break large scripts into modules for reusability.

Example:

```powershell
$age = 20
if ($age -ge 22) {
    Write-Host "Age is greater"
}

# IF-ELSEIF-ELSE example
$age = 25
if ($age -ge 18) {
    Write-Host "Adult"
} elseif ($age -ge 13) {
    Write-Host "Teenager"
} else {
    Write-Host "Child"
}

# Read content from file and greet
$data = Get-Content -Path "C:\Users\Sooraj B\Desktop\6thsem\download.txt"
Write-Output "Hello, let's welcome $data"
```


```powershell
    Write-Host "Age is greater"
}

# IF-ELSEIF-ELSE example
$age = 25
if ($age -ge 18) {
    Write-Host "Adult"
} elseif ($age -ge 13) {
    Write-Host "Teenager"
} else {
    Write-Host "Child"
}

# Read content from file and greet
$data = Get-Content -Path "C:\Users\Sooraj B\Desktop\6thsem\download.txt"
Write-Output "Hello, let's welcome $data"
```


```powershell
# Get service members
Get-Service | Get-Member | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append

# Get date members
```

powershell

CopyEdit

function Get-SystemInfo { Get-CimInstance Win32_OperatingSystem } Export-ModuleMember -Function Get-SystemInfo

5. Input, Output & Formatting

Example Input & Output:

```powershell
Get-Process -Name "brave" | Stop-Process

# Show service status
Get-Service | ForEach-Object Status | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append

# Show service display names
Get-Service | ForEach-Object DisplayName | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append

# Show event logs
Get-EventLog -List | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append

# Show only system log
Get-EventLog -List | Where-Object Log -eq "System" | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append

# Convert process list to HTML
Get-Process | ConvertTo-Html | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append

```

```
Get-EventLog -List | Where-Object Log -eq "System" | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append

# Convert process list to HTML
Get-Process | ConvertTo-Html | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append


Handles  NPM(K)    PM(K)      WS(K)     CPU(s)      Id  SI ProcessName
-------  ------    -----      -----     ------      --  -- -----------
    191      11     4380       5552       7.86    7632   0 AggregatorHost
    235      15    22448        456       7.47   20668   1 ai
    153      10     2408        148       1.28    3264   0 amdfendrsr
    300      17    16380       4508       4.53    9248   1 AppActions
    302      20     4784       1680     118.09    6456   1 AppleMobileDeviceProcess
    454      26    13028      16408      18.69   14120   1 ApplicationFrameHost
    682      17     4772       3120     220.92    3484   1 atieclxx
    200      10     1600        552       0.33    3276   0 atiesrxx
    329      28    17860        404       0.41   11780   1 backgroundTaskHost
    331      28    17804        416       0.42   15916   1 backgroundTaskHost
    329      28    17988        420       0.98   17600   1 backgroundTaskHost
    331      28    17508        408       0.77   24288   1 backgroundTaskHost
    330      28    17504        404       0.50   25212   1 backgroundTaskHost
```

powershell

CopyEdit

$name = Read-Host "Enter your name" Write-Output "Hello, $name" | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append

Formatting Example:

powershell

CopyEdit

Get-Service | Format-Table Name, Status | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append

6. Combined

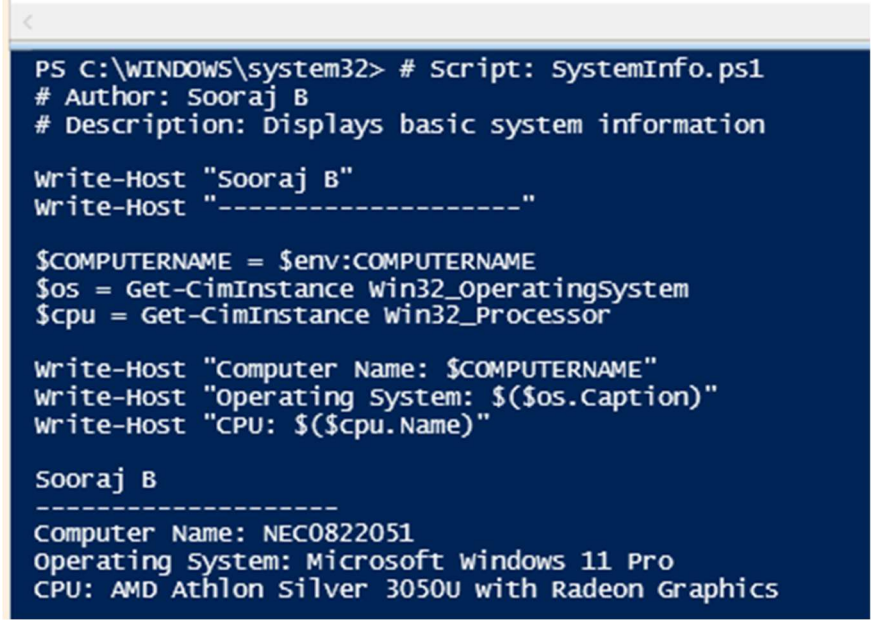
```powershell
# Script: SystemInfo.ps1
# Author: Sooraj B
# Description: Displays basic system information

Write-Host "Sooraj B"
Write-Host "--------------------"

$COMPUTERNAME = $env:COMPUTERNAME
$os = Get-CimInstance Win32_OperatingSystem
$cpu = Get-CimInstance Win32_Processor

Write-Host "Computer Name: $COMPUTERNAME"
Write-Host "Operating System: $($os.Caption)"
Write-Host "CPU: $($cpu.Name)"

```



powershell

CopyEdit

Get-WmiObject Win32_NetworkAdapterConfiguration | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append Get-NetIPConfiguration | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append Get-Service | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append Get-PSDrive -PSProvider FileSystem | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append Get-Command | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append Get-ChildItem | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append Get-Process | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append Get-Hotfix | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append Get-NetFirewallRule | Out-File "C:\Users\Sooraj B\Desktop\6thsem\download.txt" -Append

```powershell
Clear-Host
Get-CimInstance Win32_OperatingSystem
Get-Command Get-CimInstance
Get-Help Get-CimInstance
Import-Module BitsTransfer
Get-command Install-Module, Install-PSResource
Get-Module
Get-Module -ListAvailable
#Get-Command -Module <module-name>
Get-command -Module BitsTransfer
Remove-Module BitsTransfer
Get-command Get-Date, Get-Help -All | Select-object -Property Name, CommandType, Module ,PSSnapIn
function Get-Date {
    Microsoft.PowerShell.Utility\Get-Date -Format "yyyy-MM-dd HH:mm:ss"
}

# Call your custom function
Get-Date

# Remove your custom function
```

```
SystemDirectory     Organization BuildNumber RegisteredUser SerialNumber            Version
---------------     ------------ ----------- -------------- ------------            -------
C:\WINDOWS\system32              26100       Sooraj B       00331-10000-00001-AA485 10.0.26100

Verb                : Get
Noun                : CimInstance
HelpFile            : C:\WINDOWS\system32\WindowsPowerShell\v1.0\Modules\CimCmdlets\en-US\Microsoft.Management.Infrastructure.
                      CimCmdlets.dll-help.xml
PSSnapIn            :
Version             : 1.0.0.0
ImplementingType    : Microsoft.Management.Infrastructure.CimCmdlets.GetCimInstanceCommand
Definition          :
                      Get-CimInstance [-ClassName] <string> [-ComputerName <string[]>] [-KeyOnly] [-Namespace <string>]
                      [-OperationTimeoutSec <uint32>] [-QueryDialect <string>] [-Shallow] [-Filter <string>] [-Property
                      <string[]>] [<CommonParameters>]

                      Get-CimInstance -CimSession <CimSession[]> -ResourceUri <uri> [-KeyOnly] [-Namespace <string>]
                      [-OperationTimeoutSec <uint32>] [-Shallow] [-Filter <string>] [-Property <string[]>]
                      [<CommonParameters>]

                      Get-CimInstance -CimSession <CimSession[]> -Query <string> [-ResourceUri <uri>] [-Namespace <string>]
```

Final Thoughts

This day covered:

- Using **PSADT** for real deployments
- Writing **structured scripts**
- Querying system data with **WMI/CIM**
- Organizing code via **modules**
- Handling **input/output & formatting**
- Building **reusable code** for automation