## Objectives:

* Describe Desired State Configuration (DSC) and its components
* Create and deploy a DSC configuration
* Test a DSC deployment
* Correct configuration drift using DSC
* Practice using DSC

## The following virtual machines will be used in this lab:

* AcmeServer2019
* Windows10

**PreLab Setup:**

On the Windows10 virtual machine, set the PowerShell execution policy to RemoteSigned and execute the Enable-PSRemoting cmdlet from an elevated PowerShell Session. (Run As Administrator)

## Describe Desired State Configuration (DSC) and its components

Windows PowerShell Desired State Configuration (DSC) was introduced with Windows Management Framework (WMF) 4.0 and PowerShell 4. It enables automated, ‘declarative’ configuration of the specified target nodes (devices and computers) in your IT infrastructure. Device configuration is declarative. The IT administrator declares what should be installed on the computer without having to define how the required resources are installed.

For example, a server role can be installed without an administrator having to write a script to instruct the computer how to install the feature, make the appropriate configuration changes, copy the necessary files, etc.

The DSC architecture knows how to implement the changes on the node to ensure the device conforms to the desired configuration, including installing, modifying, or removing components without being instructed on how to do so. The ‘How to’ is contained within DSC Resources and is managed by the Local Configuration Manager (LCM). You do not have to know how to implement or deploy a feature when a DSC resource exists.

This configuration management system enables you to configure your system ‘As Code’. A DSC configuration is a component of Infrastructure as Code (IaC). Infrastructure as Code provides the following benefits:

* Reliable, consistent, efficient configuration and setup
* Quick and easy deployments across different environments
* Reduced risk of human error
* Enhanced security and compliance

## How does it work?

The following components are involved in PowerShell DSC implementation.

* DSC configuration scripts and resources
* Managed Object Format (MOF) files
* The Local Configuration Manager on the target computer

The DSC configuration script contains the ‘declarations’ of what the device should look like. The script will list the DSC resources (Files, Roles, services, etc.) that are to be configured and what their configuration state should be.

The DSC configuration script is then compiled into a Managed Object Format (MOF) file. The MOF file represents the configuration information in a way that can be understood by the Common Information Model (CIM). The goal of the CIM is to provide an open standard that can be applied across multiple platforms. Windows, Linux, MacOS, etc.

The MOF file is deployed to the nodes (devices or computers) specified in the configuration script. The configuration can be pushed to the computer or can be pulled from a central file share or URL

The Local Configuration Manager is the component running on the target node that receives, coordinates, and implements the configuration specified.

## DSC Resources

The DSC resource is responsible for testing, implementing, or remediating its own configuration elements. For example, if the MOF file requires a specific Server Role to be installed, the *DSC Role* resource will take over, it knows how to test for, install and configure server roles.

WMF 4.0 ships with some basic DSC resources. This introduction will use the WindowsFeature and User DSC resources.

WindowsFeature “*ResourceName”*

{

Name = “string”

Credential = PSCredential Object

Ensure = “Absent” or “Present”

IncludeAllSubFeature = $True or $False

LogPath = “string”

DependsOn = “[Resource]Name” #used to check for a dependent configuration

Source = “string” #path to the installation files if necessary

}

User “*ResourceName*”

{

UserName = “string”

Description = “string”

Disabled = $True or $False

Ensure = “Absent” or “Present”

FullName = “string”

Password = PSCredential Object

PasswordChangeNotAllowed = $True or $False

PasswordChangeRequired = $True or $False

PasswordNeverExpires = $True or $False

DependsOn = “[Resource]*ResourceName*” #used to check for a dependent resource configuration

}

See https://learn.microsoft.com/en-us/powershell/dsc/resources/resources?view=dsc-1.1 for a list of the built-in Windows PowerShell Desired State Configuration Resources.

## Create a DSC to manage a remote system

The configuration script includes the following keywords:

**Keyword:** Configuration

Used to define the desired configuration

**Syntax:** Configuration *ConfigurationName {StatementList}*

Everything relating to the named configuration will be contained within the configuration StatementList.

Within the StatementList there will be one or more Node sections that define the computers that the configuration will be applied to.

**Keyword:** Node

Used to define the desired configuration for one or more nodes.

**Syntax:** Node *NodeName* *{StatementList}*

Within the StatementList there will be one or more DSC Resource sections that describe the desired state for that specific resource.

## Sample DSC Script

Create and save the following SampleConfiguration.ps1 script to the current directory on the AcmeServer2019 virtual machine.

Configuration SampleConfiguration

{

Node "Acmeserver"

{

WindowsFeature "InstallWinBackup"

{

Ensure="Present"

Name="Windows-Server-Backup"

}

}

Node "Windows10-pc"

{

User "DisableLocalAdmin"

{

Username="CSAIT"

Disabled=$true

}

}

}

Load the Configuration block by running it in the script editor or calling the saved script file.

.\SampleConfiguration.ps1

## Listing Configurations

Example:

Get-Command –CommandType Configuration

Returns:

Displays all configurations loaded in the session.

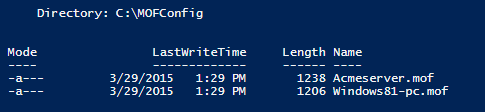
## Convert a configuration to a MOF file

Example:

New-Item C:\MOFConfig –ItemType Directory

SampleConfiguration –OutputPath C:\MOFConfig

Returns:



An MOF file will be generated for each node specified in the configuration and saved to the directory specified by the –OutputPath parameter.

## Deploy the configuration to the target nodes

Example:

Start-DscConfiguration -path C:\MOFConfig -Wait

Result:

* The MOF configuration files in the C:\MOFConfig directory will be pushed to the target nodes.
* The Acmeserver Local Configuration Manager will install the Windows server backup software.
* The Windows10 Local Configuration Manager will disable the local CSAIT Account on the Windows10 Virtual Machine.

## Test the current configuration

Example:

Test-DscConfiguration

Returns:

True

## Test the current configuration of a remote node

Example:

$node=New-CIMSession –Computername Windows10-PC

Test-DscConfiguration –CimSession $node

Returns:

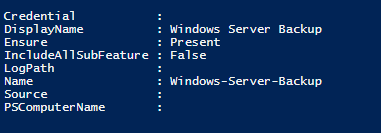
True

## Display the Current configuration

Example:

Get-DscConfiguration

Returns:



## Display the current configuration of a remote node

Example:

$node=New-CIMSession –Computername Windows10-PC

Get-DscConfiguration –CimSession $node

Returns:

A blue screen with white text

Description automatically generated

## Correct configuration drift using DSC

Login to the Windows10 virtual machine and enable the CSAIT account.

On the Acmeserver virtual machine, test the configuration.

Example:

$node=New-CIMSession –Computername Windows10

Test-DscConfiguration –CimSession $node

Returns:

False

Correct and then test the configuration again.

Example:

Restore-DscConfiguration -CimSession $node

Test-DscConfiguration –CimSession $node

By default, the local DSC engine on the target node will check for configuration drifts every 15 minutes. When the Local Configuration Manager is in “ApplyAndMonitor” mode, which is the default, it will automatically attempt to bring the node back into compliance.

## Key Observations:

* A DSC Configuration script is invoked in the same way as a function. The script is loaded in memory, then the configuration name is invoked with the required parameters.
* The DscConfiguration cmdlets use the –CimSession parameter to work with remote nodes.
* The New-CimSession cmdlet will create a CimSession for the specified computer
* The Restore-DscConfiguration cmdlet can be used to fix a configuration that has ‘drifted’ out of compliance.
* Use the Get-Command –Noun DscConfiguration command to list all the available DSC cmdlets
* Use the Get-DSCLocalConfigurationManager cmdlet to see how the Local Configuration Manager has been configured.

## Practice Exercise:

1. Create a desired state configuration script called MyPracticeDSC. The configuration script will make the following configuration changes to the Windows10 and the AcmeServer virtual machines.

* Create the following registry modifications to disable the Windows App store on the Windows10 virtual machine.

Registry value name: ‘RemoveWindowsStore’

Registry Value Data: 1

Registry Key: HKey\_Local\_Machine\Software\Policies\Microsoft\WindowsStore

* Ensure the local Administrator account is disabled on the Windows10 virtual machine.
* Ensure the Active Directory Rights Management and all its sub-features are installed on the AcmeServer Virtual machine.