Full credit is given to the above companies including the OS that this PDF file was generated!

PowerShell Get-Help on command 'Start-DscConfiguration'

PS C:\Users\wahid> Get-Help Start-DscConfiguration

NAME

Start-DscConfiguration

SYNOPSIS

Applies configuration to nodes.

SYNTAX

Start-DscConfiguration [[-Path] <System.String>] -CimSession
<Microsoft.Management.Infrastructure.CimSession[]> [-Force] [-JobName
<System.String>] [-ThrottleLimit <System.Int32>] [-Wait] [-Confirm] [-WhatIf]
[<CommonParameters>]

Start-DscConfiguration -CimSession
<Microsoft.Management.Infrastructure.CimSession[]> [-Force] [-JobName
<System.String>] [-ThrottleLimit <System.Int32>] -UseExisting [-Wait]
[-Confirm] [-WhatIf] [<CommonParameters>]

Start-DscConfiguration [[-Path] <System.String>] [[-ComputerName]
<System.String[]>] [-Credential <System.Management.Automation.PSCredential>]
[-Force] [-JobName <System.String>] [-ThrottleLimit <System.Int32>] [-Wait]

[-Confirm] [-WhatIf] [<CommonParameters>]

Start-DscConfiguration [[-ComputerName] <System.String[]>] [-Credential <System.Management.Automation.PSCredential>] [-Force] [-JobName <System.String>] [-ThrottleLimit <System.Int32>] -UseExisting [-Wait] [-Confirm] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The `Start-DscConfiguration` cmdlet applies configuration to nodes. When used with the UseExisting parameter, the existing configuration on the target computer is applied. Specify which computers that you want to apply configuration to by specifying computer names or by using Common Information Model (CIM) sessions.

By default, this cmdlet creates a job and returns a Job object. For more information about background jobs, type `Get-Help about_Jobs`. To use this cmdlet interactively, specify the Wait parameter.

Specify the Verbose parameter to see details of what the cmdlet does when it applies configuration settings.

PARAMETERS

-CimSession <Microsoft.Management.Infrastructure.CimSession[]>

Runs the cmdlet in a remote session or on a remote computer. Enter a computer name or a session object, such as the output of a New-CimSession (/powershell/module/cimcmdlets/new-cimsession)or Get-CimSession (/powershell/module/cimcmdlets/get-cimsession)cmdlet. The default is the current session on the local computer.

-ComputerName <System.String[]>

Specifies an array of computer names. This parameter restricts the

computers that have configuration documents in the Path parameter to those specified in the array.

-Credential <System.Management.Automation.PSCredential>

Specifies a user name and password, as a PSCredential object, for the target computer. To obtain a PSCredential object, use the `Get-Credential` cmdlet. For more information, type `Get-Help Get-Credential`.

-Force <System.Management.Automation.SwitchParameter>

Stops the configuration operation currently running on the target computer and begins the new Start-Configuration operation. If the RefreshMode property of the Local Configuration Manager is set to Pull , specifying this parameter changes it to Push .

-JobName <System.String>

Specifies a friendly name for a job. If you specify this parameter, the cmdlet runs as a job, and it returns a Job object.

By default, Windows PowerShell assigns the name JobN where N is an integer.

If you specify the Wait parameter, do not specify this parameter.

-Path <System.String>

Specifies a file path of a folder that contains configuration settings files. This cmdlet publishes and applies these configuration settings to computers that have settings files in the specified path. Each target node must have a settings file of the following format: `<NetBIOS Name>.mof`.

-ThrottleLimit <System.Int32>

Specifies the maximum number of concurrent operations that can be established to run the cmdlet. If this parameter is omitted or a value of `0` is entered, then Windows PowerShell calculates an optimum throttle limit for the cmdlet based on the number of CIM cmdlets that are running

on the computer. The throttle limit applies only to the current cmdlet, not to the session or to the computer.

-UseExisting <System.Management.Automation.SwitchParameter>
Indicates that this cmdlet applies the existing configuration. The configuration can exist on the target computer by enactment using `Start-DscConfiguration` or by publication using the Publish-DscConfiguration cmdlet.

Before you specify this parameter for this cmdlet, review the information in What's New in Windows PowerShell 5.0 (/powershell/scripting/whats-new/what-s-new-in-windows-powershell-50).

-Wait <System.Management.Automation.SwitchParameter>
Indicates that the cmdlet blocks the console until it finishes all configuration tasks.

If you specify this parameter, do not specify the JobName parameter.

-Confirm <System.Management.Automation.SwitchParameter>
Prompts you for confirmation before running the cmdlet.

-WhatIf <System.Management.Automation.SwitchParameter>
Shows what would happen if the cmdlet runs. The cmdlet is not run.

<CommonParameters>
This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https:/go.microsoft.com/fwlink/?LinkID=113216).

---------- Example 1: Apply configuration settings ----------

```
Start-DscConfiguration -Path "C:\DSC\Configurations\"
```

This command applies the configuration settings from `C:\DSC\Configurations` to the every computer that has settings in that folder. The command returns Job objects for each target node deployed to.

Example 2: Apply configuration settings and wait for configuration to complete

```
Start-DscConfiguration -Path "C:\DSC\Configurations\" -Wait -Verbose
```

This command applies the configuration from `C:\DSC\Configurations` to the local computer. The command returns Job objects for each target node deployed to, in this case, just the local computer. This example specifies the Verbose parameter. Therefore, the command sends messages to the console as it proceeds. The command includes the Wait parameter. Therefore, you cannot use the console until the command finishes all configuration tasks.

Example 3: Apply configuration settings by using a CIM session

```
$Session = New-CimSession -ComputerName "Server01" -Credential
ACCOUNTS\PattiFuller
Start-DscConfiguration -Path "C:\DSC\Configurations\" -CimSession $Session
```

This example applies configuration settings to a specified computer. The example creates a CIM session for a computer named Server01 for use with the cmdlet. Alternatively, create an array of CIM sessions to apply the cmdlet to multiple specified computers.

The first command creates a CIM session by using the `New-CimSession` cmdlet, and then stores the CimSession object in the `$Session` variable. The command prompts you for a password. For more information, type `Get-Help NewCimSession`.

The second command applies the configuration settings from `C:\DSC\Configurations` to the computers identified by the CimSession objects

stored in the `$Session` variable. In this example, the $Session variable contains a CIM session only for the computer named Server01. The command applies the configuration. The command creates Job objects for each configured computer.

REMARKS

To see the examples, type: "get-help Start-DscConfiguration -examples".

For more information, type: "get-help Start-DscConfiguration -detailed".

For technical information, type: "get-help Start-DscConfiguration -full".

For online help, type: "get-help Start-DscConfiguration -online"