# Table of Contents

Lab Overview - HOL-1911-05-SDC - vSphere Automation - PowerCLI ................................. 2  
Lab Guidance .................................................................................................................. 3  
Module 1 - Introduction to VMware PowerCLI (30 minutes) ........................................ 9  
  Introduction.................................................................................................................. 10  
  Installing and Upgrading VMware PowerCLI ............................................................. 12  
  VMware PowerCLI Help ............................................................................................. 18  
  PowerCLI Syntax, Pipelines, Wildcards, Variables, and Parameters ......................... 22  
  Conclusion.................................................................................................................... 27  
Module 2 - Automate Configuration Activities for vCenter Server (30 Minutes) .......... 29  
  Introduction................................................................................................................ 30  
  Connect to the vCenter Server Using VMware PowerCLI ......................................... 32  
  Create a Datacenter in vCenter Using VMware PowerCLI in Windows .................... 34  
  Connect to vCenter and Create a Datacenter Using VMware PowerCLI in Linux ........ 37  
  Get and Set Cluster Settings ....................................................................................... 44  
  Manage Resource Pools with VMware PowerCLI ..................................................... 52  
  Create and Manage Folders with VMware PowerCLI ............................................. 56  
  Conclusion.................................................................................................................... 62  
Module 3 - Automate Configuration Activities for ESXi Hosts (15 minutes) ................. 64  
  Introduction................................................................................................................ 65  
  Connect to vCenter Using VMware PowerCLI in Linux ........................................... 66  
  List ESXi Hosts Within the vCenter Server ............................................................... 71  
  Adding Hosts and Managing Status ............................................................................ 73  
  Get Datastore Information and Rescan HBAs and VMFS Datastores ....................... 76  
  Conclusion.................................................................................................................... 80  
Module 4 - Automate Virtual Machine Management (30 minutes) ............................... 81  
  Introduction................................................................................................................ 82  
  Connect to the vCenter Server Using VMware PowerCLI ......................................... 84  
  Working With VM Templates ...................................................................................... 86  
  Managing Virtual Machine Configuration ................................................................. 89  
  Managing Hard Disk Configuration ............................................................................ 91  
  Managing Virtual Machine Power States .................................................................... 93  
  Managing VM Snapshots ............................................................................................ 96  
  Conclusion.................................................................................................................... 100
Lab Overview - HOL-1911-05-SDC - vSphere Automation - PowerCLI
Lab Guidance

Note: It may take more than 90 minutes to complete this lab. You should expect to only finish 2-3 of the modules during your time. The modules are independent of each other so you can start at the beginning of any module and proceed from there. You can use the Table of Contents to access any module of your choosing.

The Table of Contents can be accessed in the upper right-hand corner of the Lab Manual.

You still haven’t learned PowerCLI? What are you waiting for? VMware's PowerCLI is a great tool enabling rapid and easy interaction with your vSphere environment and beyond. This introductory class will enable you to go hands-on with VMware PowerCLI. Gain familiarity with the tool, and then dive deeper into the functionalities available in this exciting product.

Lab Module List:

- **Module 1 - Introduction to PowerCLI** (30 minutes) (Basic) In this module, you will learn about VMware's PowerCLI command-line tool. Along with an overview of PowerCLI, we will discuss installation and configuration of PowerCLI in both Windows and Linux.
- **Module 2 - Automate Configuration Activities for vCenter Server** (30 minutes) (Basic) In this module, you will learn how to Automate configuration activities for vCenter Server using PowerCLI.
- **Module 3 - Automate Configuration Activities for ESXi Hosts** (15 minutes) (Basic) In this module, you will learn how to Automate configuration activities for ESXi hosts using PowerCLI.
- **Module 4 - Automate Virtual Machine Management** (30 minutes) (Basic) In this module, you will learn how to automate virtual machine management using PowerCLI.

Lab Captains:

- Module 1 - Howard Shoobe, Professional Services, USA
- Module 2 - Howard Shoobe, Professional Services, USA
- Module 3 - Howard Shoobe, Professional Services, USA
- Module 4 - Howard Shoobe, Professional Services, USA

This lab manual can be downloaded from the Hands-on Labs Document site found here:

http://docs.hol.vmware.com
This lab may be available in other languages. To set your language preference and have a localized manual deployed with your lab, you may utilize this document to help guide you through the process:


**Location of the Main Console**

1. The area in the RED box contains the Main Console. The Lab Manual is on the tab to the Right of the Main Console.
2. A particular lab may have additional consoles found on separate tabs in the upper left. You will be directed to open another specific console if needed.
3. Your lab starts with 90 minutes on the timer. The lab can not be saved. All your work must be done during the lab session. But you can click the **EXTEND** to increase your time. If you are at a VMware event, you can extend your lab time twice, for up to 30 minutes. Each click gives you an additional 15 minutes. Outside of VMware events, you can extend your lab time up to 9 hours and 30 minutes. Each click gives you an additional hour.

**Alternate Methods of Keyboard Data Entry**

During this module, you will input text into the Main Console. Besides directly typing it in, there are two very helpful methods of entering data which make it easier to enter complex data.
Click and Drag Lab Manual Content Into Console Active Window

You can also click and drag text and Command Line Interface (CLI) commands directly from the Lab Manual into the active window in the Main Console.

Accessing the Online International Keyboard

You can also use the Online International Keyboard found in the Main Console.

1. Click on the Keyboard Icon found on the Windows Quick Launch Task Bar.
Click once in active console window

In this example, you will use the Online Keyboard to enter the "@" sign used in email addresses. The "@" sign is Shift-2 on US keyboard layouts.

1. Click once in the active console window.
2. Click on the Shift key.

Click on the @ key

1. Click on the "@ key".

Notice the @ sign entered in the active console window.
Activation Prompt or Watermark

When you first start your lab, you may notice a watermark on the desktop indicating that Windows is not activated.

One of the major benefits of virtualization is that virtual machines can be moved and run on any platform. The Hands-on Labs utilizes this benefit and we are able to run the labs out of multiple datacenters. However, these datacenters may not have identical processors, which triggers a Microsoft activation check through the Internet.

Rest assured, VMware and the Hands-on Labs are in full compliance with Microsoft licensing requirements. The lab that you are using is a self-contained pod and does not have full access to the Internet, which is required for Windows to verify the activation. Without full access to the Internet, this automated process fails and you see this watermark.

This cosmetic issue has no effect on your lab.

Look at the lower right portion of the screen
Please check to see that your lab is finished all the startup routines and is ready for you to start. If you see anything other than "Ready", please wait a few minutes. If after 5 minutes you lab has not changed to "Ready", please ask for assistance.
Module 1 - Introduction to VMware PowerCLI (30 minutes)
Introduction

In this module, we will cover the basics of using VMware's PowerCLI command-line tool. We will go over how to install or upgrade it, how to use help from within the tool, the proper syntax, pipelines, wildcards, variables, and parameters, and finally how to launch the tool.

This Module contains the following lessons:

- **Installing and Upgrading PowerCLI Considerations**: We go over some of the requirements to install and update VMware PowerCLI.
- **VMware PowerCLI Help**: We will cover the Get-Help command as well as how to update the help content from Microsoft.
- **PowerCLI Syntax, Pipelines, Wildcards, Variables, and Parameters**: We learn how to use proper Syntax as well as the use of Pipelines, Wildcards, Variables, and Parameters in PowerCLI.
- **Getting Started With PowerShell and PowerCLI**: We learn how to launch PowerShell, PowerCLI, and then open a few panes to make it easier to work with.

Introducing VMware PowerCLI

VMware PowerCLI offers an excellent command-line alternative to the point-and-click based administration of the vSphere Web Clients. As virtualization has become mainstream and deployments begin to sprawl, the simple commands of VMware PowerCLI allows faster administration by executing tasks against large groups of objects in the virtual environment.

VMware PowerCLI follows a very logical pattern, it can be quickly adopted and understood, making it the first choice for many vSphere administrators. It also combines extensibility to allow users to build their own functions and modules to solve specific problems not addressed by out-of-box functionalities present in the vCenter Graphical User Interface (GUI) console.

The focus of this lab session will be on VMware PowerCLI in a vSphere environment.

Overview/Benefits

VMware helps enterprise IT overcome cloud management challenges with the following differentiators:

- **Achieving Fastest Time-to-Cloud Value**: extensive out-of-the-box capabilities, extensible architecture and ability to leverage existing investments, enables you to create scalable private, public and desktop cloud services in just days
- **Rapid Ecosystem Integration**: a full spectrum of extensibility options that empowers IT personnel to enable, adapt, and extend their cloud to work within
their existing IT infrastructure and processes, thereby eliminating expensive service engagements while reducing risk

- **Embracing Consumerization of IT**: unifies disparate systems and processes into a highly personalized, self-service experience governed by user and organizational awareness, ensuring the exact needs of the user are met in the context of the business

- **Boosting IT Efficiency**: VMware's advanced self-service and automation capabilities lead to a significant reduction in Operational Expenditures (OpEx), while organizationally aware governance enables a multi-tenant infrastructure, improving hardware utilization and eliminating Capital Expenditures (CapEx)

- **Increasing Business Agility**: enterprise IT is empowered to quickly enable cloud services so that the business can react quickly to changing market demands and capture market share

- **Hybrid Cloud Automation**: agnostic provisioning of hardware platforms, operating systems, hypervisors, management tools, and public clouds; vRealize Automation allows customers to rapidly stand up cloud services to deliver quick time to value from your IT investments
Installing and Upgrading VMware PowerCLI

In this entire lesson, we will discuss the considerations and associated steps of installing and updating VMware PowerCLI.

NOTE: In this particular module, we WILL NOT be performing ANY PowerShell or VMware PowerCLI commands in the Hands On Lab (HOL) lab environment that we will working in!

Supported Operating Systems:

You can install VMware PowerCLI on supported Windows, Linux and MacOS operating systems. You can run guest cmdlets against virtual machines on which supported guest operating systems are installed.

• PowerCLI Local Operating Systems:

   For a list of operating systems on which you can install VMware PowerCLI, see the Compatibility Matrixes for VMware PowerCLI located here.

• PowerCLI Guest Operating Systems:

   You can run VMware PowerCLI guest cmdlets against virtual machines with supported guest operating systems. For a list of operating systems on which you can install VMware PowerCLI, see the Compatibility Matrixes for VMware PowerCLI located here.

   Note: Guest cmdlets are not compatible with IPv6 environments.

Supported VMware Products:

You can use the PowerCLI modules to manage all supported VMware products. For a list of VMware products with which VMware PowerCLI is compatible, see VMware Product Interoperability Matrixes located here.

Supported Windows PowerShell Versions:

VMware PowerCLI is compatible with multiple versions of Windows PowerShell and PowerShell core (for Linux and MacOS). For a list of PowerShell versions with which VMware PowerCLI is compatible, see Compatibility Matrixes for VMware PowerCLI located here.

Prerequisites for Installing and Running PowerCLI:
Before installing and running VMware PowerCLI, verify that you have installed the required software on the same machine. For a list of software that you need if you want to work with VMware PowerCLI, see the Installation Prerequisites section of the Compatibility Matrixes for VMware PowerCLI located here.

Installing VMware PowerCLI

VMware PowerCLI can be installed in either Windows PowerShell or PowerShell core (for Linux and MacOS). Once PowerShell is installed, VMware PowerCLI can be installed by running a PowerShell command. You can install all official modules with a single command, or install modules individually. The VMware PowerCLI modules are available on the PowerShell Gallery Web site. When you run Install-Module from the Windows PowerShell prompt, the command downloads and installs the specified module. For a list of available VMware PowerCLI modules, see the PowerShell Gallery Web site located here.

Prerequisites:

- Before installing VMware PowerCLI, see the prerequisites that we covered at the beginning of this module.
- Verify that you have uninstalled VMware PowerCLI 6.5.x or earlier from your system.
- Verify that you have registered the PowerShell Gallery as a local repository.
- Verify that your system is connected to the Internet.
- If you use Windows PowerShell 3.0 or 4.0, install the PowerShell Get and Package Management modules from the PowerShell Gallery Web site.

**NOTE:** In this module, we WILL NOT be performing ANY PowerShell or VMware PowerCLI commands in the Hands On Lab (HOL) lab environment that we will working in!

Procedure:

- For MacOS or Linux

  The first step is to install PowerShell core. Instructions for a variety of Linux variants can be found here.

- Register the Microsoft RedHat repository.

  curl https://packages.microsoft.com/config/rhel/7/prod.repo | sudo tee /etc/yum.repos.d/microsoft.repo

- Install PowerShell

  sudo yum install -y powershell

Click below to see a video example of PowerShell core being installed on CentOS.
• Once PowerShell core is installed, it is started using the command:

  pwsh

• For Windows, PowerShell is typically installed by default and can be opened by clicking the PowerShell Icon.

  ![PowerShell Icon](image)

• Once the PowerShell command line has been opened, all VMware PowerCLI modules are installed by running the following command.

  Install-Module -Name VMware.PowerCLI -Scope CurrentUser

An example of the PowerCLI installation in Linux can be seen below.
Allow Execution of Scripts

If you want to run scripts and load configuration files with VMware PowerCLI, you must set the execution policy of Windows PowerShell to "RemoteSigned".

For security reasons, Windows PowerShell supports an execution policy feature. It determines whether scripts are allowed to run and whether they must be digitally signed. By default, the execution policy is set to Restricted, which is the most secure policy. For more information about the execution policy and script digital signing in Windows PowerShell, run Get-Help About_Signing. You can change the execution policy by using the Set-ExecutionPolicy cmdlet.

**NOTE:** We will NOT be uninstalling or installing versions of PowerCLI in this lab. The above information on removing it was for INFORMATIONAL purposes only!

- Open the Windows PowerShell console.

  Run Set-ExecutionPolicy RemoteSigned
Updating PowerCLI Considerations

When we need to upgrade VMware PowerCLI to a new version, there are some steps that should be done prior to installing the new version of VMware PowerCLI. If VMware PowerCLI was installed using a Microsoft .MSI file, we will need to uninstall the existing instance of Windows PowerShell prior to upgrading VMware PowerCLI. It is also worth checking to ensure the “PowerCLI” folder has been removed from the following directory (C:\Program Files (x86)\VMware\Infrastructure\).

NOTE: We will NOT be uninstalling or installing versions of PowerCLI in this lab. The above information on removing it was for INFORMATIONAL purposes only!

- Open the Windows PowerShell console.
- Uninstall the existing version of the module. Do not run this command in the lab.
  
  Get-Module VMware.Module_Name | Uninstall-Module -Force

- Install the new version of the module.

  Install-Module VMware.Module_Name
Windows PowerShell - Online Update

For the online install, start by confirming access to Windows PowerShell Gallery and being able to find the VMware PowerCLI module. This can be done by running the following command.

*NOTE:* DO NOT run this command in the HOL lab environment! This is for informational purposes only.

```
Update-Module -Name VMware.PowerCLI
```
VMware PowerCLI Help

In this lesson, we will discuss how to get help with the VMware PowerCLI commands. It has a built-in help to assist with proper syntax of all the VMware PowerCLI commands.

**NOTE: In this particular module, we WILL NOT be performing ANY PowerShell or VMware PowerCLI commands in the Hands On Lab (HOL) lab environment that we will working in!**

**Get-PowerCLIHelp**

If you are looking for help with VMware PowerCLI, you can type the command "Get-PowerCLIHelp". It will then open up your default web browser and connect you to this [web page](https://www.vmware.com/support/developer/PowerCLI/) providing your computer has an internet connection.
Update Help From Within VMware PowerCLI

NOTE: DO NOT run this command in the HOL lab environment that we are currently working in. This is meant for informational purposes only!

We recommend that every once in a while, you update the help content to ensure you have the latest information. I would especially do it after any VMware PowerCLI updates that you install. To get help updated, simple type "Update-Help" to get the latest content from Microsoft.
Help From Within the VMware PowerCLI Command-Line Tool

NOTE: DO NOT run this command in the HOL lab environment that we are currently working in. This is meant for informational purposes only!

If you need help while working inside the VMware PowerCLI command-line tool itself, that is also available. To get help on any command, you can simple type "Get-Help Command_Name" to get assistance on the proper syntax of that specific command.

1. Type the following command in the upper pane of VMware PowerCLI.
   
   Get-Help Get-VM

1. Click on the "Run Script" icon in the menu to execute the command.

2. We now see that it provides us help on the command that we specifically ask for which was the Get-VM command in this case.
Help From Within the VMware PowerCLI Command-Line Tool (continued)

NOTE: DO NOT run this command in the HOL lab environment that we are currently working in. This is meant for informational purposes only!

To get help on any command, you can simply type "Get-Help Command_Name -Full" to get detailed help on the command.

1. Type the following command in the upper pane of VMware PowerCLI.

   Get-Help Get-VM -Full

1. Click on the "Run Script" icon in the menu to execute the command.
2. We now see that it provides us a more detailed help when we add the "-Full" attribute to it.
PowerCLI Syntax, Pipelines, Wildcards, Variables, and Parameters

In this lesson, we will cover some of the core basics of using VMware PowerCLI. These are using the proper VMware PowerCLI Syntax, the use of the Pipe ( | ) operator, the use of Wildcards (*), and some Common Parameters.

**NOTE:** In this particular module, we WILL NOT be performing ANY PowerShell or VMware PowerCLI commands in the Hands On Lab (HOL) lab environment that we will working in!

Proper PowerShell/PowerCLI Syntax

PowerShell and VMware PowerCLI cmdlets use a consistent verb-noun structure, where the verb represents the action and the noun represents the object to operate on. The cmdlets follow consistent naming patterns, ensuring that construction of a command is easy if you know the object that you want to work with. All command categories take parameters and arguments. A parameter starts with a hyphen and is used to control the behavior of the command. An argument is a data value consumed by the command.

A simple command has the following syntax:

Verb-Noun -parameter1 argument1 -parameter2 argument2, argument3

Proper Syntax for Quotation Marks Usage

Another important thing to know in regards to using proper Windows PowerShell and VMware PowerCLI syntax to ensure success is to know when you do and do not have to use quotation marks ("My VM") around names. When you are using any of the commands and you are giving the name of a vCenter object, you **DO NOT** need quotation marks when there is no space between the words in the name. If you have a name that has a space in the name, you **MUST** put the full name within quotation marks.

**Examples:**

1. **Quotation marks NOT needed**
   
   Get-VM -Name MyVM

2. **Quotation marks MUST be used:**
   
   Get-VM -Name "My VM"
Get-VM -Name "My VM"

**The Pipe ( | ) Operator**

A pipeline is a series of commands separated by the "Pipe" Operator ( | ). Each command in the pipeline receives an object from the previous command, performs some operation on it, and then passes it along to the next command in the pipeline. Objects are output from the pipeline as soon as they become available. You can type a pipeline on a single line, or spread it across multiple lines. You can cycle backwards through command history using the up arrow, so it is easier to repeat pipelines if you type them on a single line.

Get-Cluster -Name RegionA01-COMP01 | Get-VMHost

This example uses the "Get-Cluster" command to connect to the **RegionA01-COMP01** cluster, and then takes the output of that and uses it as the input to the "Get-VMHost" command. The end result returns a list of all ESXi hosts in the RegionA01-COMP01 cluster.
PowerShell has a number of pattern-matching operators named wildcards that you can use to substitute one or more characters in a string, or substitute the complete string. All wildcard expressions can be used with the VMware PowerCLI cmdlets. For example, you can view a list of all files with a .txt extension by running `Get-ChildrenItem *.txt`. In this case, the asterisk * operator matches any combination of characters. With wildcard patterns you can indicate character ranges as well. For example, to view all files that start with the letter "S" or "T" and have a "txt" extension, you can run `Get-ChildrenItem [st]*.txt`.

You can use the question mark (?) wildcard to match any single character within a sequence of characters. For example, to view all .txt files with names that consist of string and one more character at the end, run `Get-ChildrenItem string?.txt`.

```
Get-VM -Name *.corp.local
```

In the example provided in the screen capture as well as the below command, we used the "Get-VM" command and then used the argument "-Name *.corp.local". So we searched for any virtual machine that had a Fully Qualified Domain Name (FQDN) that is on the "corp.local" domain.
Using Variables

When administrators use the same commands over and over again, it can be cumbersome to retype the same thing over and over again. This is especially the case with more advanced commands much longer that what we will see in this lab. So to help reduce the amount of commands they have to type out, they will sometimes create their own "variables" which you can think of as creating a shortcut.

The below commands and in the screen shot, we see how to create a variable to use over and over again to shorten the amount of typing. The first line of commands is where we create the variable name and what it represents. So we see we created the variable 

\[\text{`$allvms = Get-VM -Name "*.corp.local"`}\]

That essentially means get all VMs that have a Fully Qualified Domain Name (FQDN) that is on the "corp.local" domain.

\[\text{PS C:\> `$allvms = Get-VM -Name "*.corp.local"`}\]

Then the next line of code is what we would type in anytime in the future to reduce the amount of typing. So we use the "Get-VM" command to get all VMs and with the Fully Qualified Domain Name (FQDN) that is on the "corp.local" domain by using our newly created variable "$allvms".

\[\text{Get-VM -Name `$allvms`}\]

In a production environment, administrators will typically be typing in much longer commands and many lines of it as well. So in the production world, using the variable will prove to be more fruitful than the simple example of one we have shown here.
VMware PowerCLI Common Parameters

The Windows PowerShell engine retains a set of parameter names, referred to as common parameters. All PowerShell cmdlets, including the PowerCLI cmdlets, support them. Some of the Windows PowerShell common parameters are Verbose, Debug, ErrorAction, ErrorVariable, OutVariable, and OutBuffer. For a full list of the common parameters and more details on their usage, run Get-Help about_CommonParameters.


- **WhatIf** - Displays the effects of a command without running it.
- **Confirm** - Prompts for confirmation before running a command that stops a program or service, or deletes data.
Conclusion

Congratulations on completing Module 1!

In this module, we learned how to Automate configuration activities for the vCenter server using VMware PowerCLI.

The Module contained the following lessons:

- **Installing and Upgrading PowerCLI Considerations**: We go over some of the requirements to install and update PowerCLI.
- **VMware PowerCLI Help**: We will cover the Get-Help command as well as how to update the help content from Microsoft.
- **PowerCLI Syntax, Pipelines, Wildcards, Variables, and Parameters**: We learn how to use proper Syntax as well as the use of Pipelines, Wildcards, Variables, and Parameters.

VMware PowerCLI Resource Links:

Below are a few additional resources available to help you get more familiar with VMware's command-line tool VMware PowerCLI.

- **VMware PowerCLI Blog**: [https://blogs.vmware.com/PowerCLI/](https://blogs.vmware.com/PowerCLI/)
- **VMware PowerCLI - User's Guide**:
- **VMware PowerCLI - Compatibility Matrix**: [https://vdc-download.vmware.com/vmwb-repository/dcr-public/ed1e0b27-dfe1-41d4-8f6e-74c4288466c1/5aef2d74-c944-469b-96e6-9f7e3779e67d/powercli1000-compat-matrix.html](https://vdc-download.vmware.com/vmwb-repository/dcr-public/ed1e0b27-dfe1-41d4-8f6e-74c4288466c1/5aef2d74-c944-469b-96e6-9f7e3779e67d/powercli1000-compat-matrix.html)
- **VMware PowerCLI - GitHub Example Scripts**: [https://github.com/vmware/PowerCLI-Example-Scripts/tree/master/Modules/VMware.VMEncryption](https://github.com/vmware/PowerCLI-Example-Scripts/tree/master/Modules/VMware.VMEncryption)

**OPTIONAL: How to End the Lab**

You can now continue to the next module by clicking forward, or use the Table of Contents to skip to another desired Module.
If you don't plan to take any more modules in the lab at this time, click on the **END** button.

**Note:** If you end your lab, you will need to re-register for the lab in order to take any other modules.
Module 2 - Automate Configuration Activities for vCenter Server (30 Minutes)
Introduction

In this module, you will learn how to Automate configuration activities for the vCenter server using VMware's PowerCLI command-line tool.

This Module contains the following lessons:

- **Connect to the vCenter Server Using VMware PowerCLI**: We learn how to connect to a vCenter server using VMware PowerCLI, this is the first step which needs to be done to start creating or modifying vCenter server objects.
- **Create a Datacenter in vCenter Using PowerCLI in Windows**: We learn how to create a datacenter using Windows PowerShell and VMware PowerCLI.
- **Connect to vCenter and Create a Datacenter Using PowerCLI in Linux**: We learn how to create a datacenter using PowerShell Core and VMware PowerCLI in Linux.
- **Get and Set Cluster Settings**: We learn how get cluster advanced properties, modify them, as well as create a cluster using VMware PowerCLI.
- **Manage Resource Pools with VMware PowerCLI**: We learn how to manage Resource Pools using VMware PowerCLI.
- **Create and Manage Folders with VMware PowerCLI**: We learn how to create and manage folders using VMware PowerCLI.

Introducing VMware PowerCLI

VMware PowerCLI offers an excellent command-line alternative to the point-and-click based administration of the vSphere Web Clients. As virtualization has become mainstream and deployments begin to sprawl, the simple commands of VMware PowerCLI allows faster administration by executing tasks against large groups of objects in the virtual environment.

VMware PowerCLI follows a very logical pattern, it can be quickly adopted and understood, making it the first choice for many vSphere administrators. It also combines extensibility to allow users to build their own functions and modules to solve specific problems not addressed by out-of-box functionalities present in the vCenter Graphical User Interface (GUI) console.

The focus of this lab session will be on VMware PowerCLI in the context of managing the vCenter Server.

Overview/Benefits

VMware helps enterprise IT overcome cloud management challenges with the following differentiators:
• **Achieving Fastest Time-to-Cloud Value**: extensive out-of-the-box capabilities, extensible architecture and ability to leverage existing investments, enables you to create scalable private, public and desktop cloud services in just days

• **Rapid Ecosystem Integration**: a full spectrum of extensibility options that empowers IT personnel to enable, adapt, and extend their cloud to work within their existing IT infrastructure and processes, thereby eliminating expensive service engagements while reducing risk

• **Embracing Consumerization of IT**: unifies disparate systems and processes into a highly personalized, self-service experience governed by user and organizational awareness, ensuring the exact needs of the user are met in the context of the business

• **Boosting IT Efficiency**: VMware's advanced self-service and automation capabilities lead to a significant reduction in Operational Expenditures (OpEx), while organizationally aware governance enables a multi-tenant infrastructure, improving hardware utilization and eliminating Capital Expenditures (CapEx)

• **Increasing Business Agility**: enterprise IT is empowered to quickly enable cloud services so that the business can react quickly to changing market demands and capture market share

• **Hybrid Cloud Automation**: agnostic provisioning of hardware platforms, operating systems, hypervisors, management tools, and public clouds; vRealize Automation allows customers to rapidly stand up cloud services to deliver quick time to value from your IT investments
Connect to the vCenter Server Using VMware PowerCLI

This lesson helps define the steps which needs to be performed to connect to a vCenter server. To execute any cmdlet in VMware PowerCLI, you need to connect to the vCenter using the `Connect-VIServer` cmdlet.

By default, VMware PowerCLI cmdlets run on the vCenter Server systems to which you are already connected. When you connect to a vCenter Server system by using the `Connect-VIServer` cmdlet, the server connection is stored in the `$DefaultVIServers` array variable. This variable contains all connected servers for the current VMware PowerCLI session. To remove a server from the `$DefaultVIServers` variable, you can either use `Disconnect-VIServer` to close all active connections to this server, or modify the value of `$DefaultVIServers` manually.

Start PowerShell

To begin using PowerCLI, we must start Windows PowerShell. As discussed in the previous module, PowerCLI has already been installed.

1. Double-click on the Windows PowerShell icon on the desktop.
Connect to the vCenter Server

Next, we must connect to our vCenter server. You need to use the `Connect-VIServer` cmdlet.

1. Type in the following command at the command line and press **ENTER**.

   ```powershell
   Connect-VIServer vcsa-01a.corp.local -User administrator@corp.local -Password VMware1!
   ```

2. When you have successfully logged into the vCenter server, a prompt will be displayed that shows the server you are connected to, using Port 443, and the user account in the output pane.
Create a Datacenter in vCenter Using VMware PowerCLI in Windows

In this lesson we will explore steps to create a datacenter object in vCenter.

vSphere objects are used to create virtual datacenters. The Datacenter is at the root of vSphere and allows the clusters and other host infrastructures to be placed inside the virtual datacenter. Most VMware administrators use datacenter objects in vSphere to represent a physical site and datacenter objects are used to define the boundary where the infrastructure exists.

Search for Existing Datacenters

VMware PowerCLI has cmdlets which allow you to search for existing datacenters. To start, let’s run the Get-Datacenter cmdlet with no additional parameters to see whether there are any existing datacenter objects in vCenter.

1. Type in the following command at the command line and press ENTER.

   ```
   Get-Datacenter
   ```

2. You will see that the command returns the name of the existing RegionA01 datacenter.
Create a New Datacenter

Now let's create a new datacenter using VMware PowerCLI and call it "HOL_DC". To configure a new datacenter, we will use the cmdlet **New-Datacenter**. We also need to provide a name for the new datacenter and the folder location of where we want it created. We will be placing the new datacenter into the "HOL" folder that we just created in the previous step. We will also use `-NoRecursion` to tell the **New-Datacenter** cmdlet to create the datacenter object at the root level of the vCenter server.

1. Our first step will be to create a folder called "HOL" in which we will create our datacenter object. The new folder object will be stored in a variable named `$folder`. Type in the following command at the command line and press ENTER.

   ```powershell
   $folder = Get-Folder -NoRecursion | New-Folder -Name HOL -Verbose
   ```

   If you do not wish to place your datacenter in a folder, the above command can be replaced with the command below. This will store the vCenter's root folder object in the `$folder` variable.

   ```powershell
   $folder = Get-Folder -NoRecursion
   ```

2. Now that our folder is created, we will use the **New-Datacenter** cmdlet to create our HOL_DC datacenter. Type in the following command at the command line and press ENTER.

   ```powershell
   New-Datacenter -Location $folder -Name HOL_DC -Verbose
   ```
New-Datacenter -Location $folder -Name HOL_DC -Verbose

Confirm Creation of the HOL-DC datacenter

We are again going to run the Get-Datacenter cmdlet with no additional parameters to see what datacenters exist in the vCenter Server.

1. Type in the following command at the command line and press ENTER.

   Get-Datacenter

2. You will see that the command returns the name of the existing RegionA01 datacenter as well as the newly created HOL_DC datacenter.
Connect to vCenter and Create a Datacenter Using VMware PowerCLI in Linux

In this lesson we will explore steps to create a datacenter object in vCenter.

vSphere objects are used to create virtual datacenters. The Datacenter is at the root of vSphere and allows the clusters and other host infrastructures to be placed inside the virtual datacenter. Most VMware administrators use datacenter objects in vSphere to represent a physical site and datacenter objects are used to define the boundary where the infrastructure exists.

SSH to a Linux VM

To begin using PowerCLI, we must start SSH into a Linux VM with PowerShell and PowerCLI already installed. As discussed in previous modules, we have prepared a virtual machine, app-01a.corp.local with both PowerShell and PowerCLI already installed.
1. Double-click on the **Putty** icon on the desktop.

2. Enter **app-01a.corp.local** in the Host Name (or IP address) box
3. Click Open

4. Click Yes in the PuTTY Security Alert window.
4. Add the server's host key to the registry by clicking **Yes**.

5. Enter the username **hol**

6. Enter the password **VMware1!**
Start PowerShell

1. Type in the following command and press Enter

   pwsh

Set PowerCLI Configuration to Ignore Invalid Certificates
1. Type in the following command and press **Enter**

```
Set-PowerCLIConfiguration -InvalidCertificateAction Ignore
```

2. Confirm the action for all by typing **A** and **Enter**

### Connect to the vCenter Server

Next, we must connect to our vCenter server. You need to use the `Connect-VIServer` cmdlet.

1. Type in the following command at the command line and press **ENTER**.

```
Connect-VIServer 192.168.110.22 -User administrator@corp.local -Password VMware1!
```

Due to lab constraints, this command may not succeed the first time. If you receive an error, please execute the command a second time.

2. When you have successfully logged into the vCenter server, a prompt will be displayed that shows the server you are connected to, using Port 443, and the user account in the output pane.
**Create a New Datacenter**

Now let's create a new datacenter using VMware PowerCLI and call it "HOL_DC". To configure a new datacenter, we will use the cmdlet `New-Datacenter`. We also need to provide a name for the new datacenter and the folder location of where we want it created. We will be placing the new datacenter into the "HOL" folder that we just created in the previous step. We will also use `-NoRecursion` to tell the `New-Datacenter` cmdlet to create the datacenter object at the root level of the vCenter server.

1. Our first step will be to create a folder called "HOL" in which we will create our datacenter object. The new folder object will be stored in a variable named `$folder`. Type in the following command at the command line and press ENTER.

```
$folder = Get-Folder -NoRecursion | New-Folder -Name HOL -Verbose
```

2. Now that our folder is created, we will use the `New-Datacenter` cmdlet to create our HOL_DC datacenter. Type in the following command at the command line and press ENTER.

```
New-Datacenter -Location $folder -Name HOL-DC -Verbose
```

If you completed the prior lab chapters performing this action using Windows PowerShell and PowerCLI, you will notice that the commands for PowerCLI are the same in both Windows and Linux.
Confirm the Creation of the HOL-DC Datacenter

We are again going to run the **Get-Datacenter** cmdlet with no additional parameters to see what datacenters exist in the vCenter Server.

1. Type in the following command at the command line and press **ENTER**.

   Get-Datacenter

2. You will see that the command returns the name of the existing **RegionB01** datacenter as well as the newly created **HOL_DC** datacenter.
Get and Set Cluster Settings

In this lesson, we will work on getting information from a cluster as well as make setting changes.

Clusters are the basis for everything that is great within vSphere. Clusters are the level where individual resources become pooled and shared for virtual machines. Clusters allow all higher-level functionality within vSphere, such as an automatic restart after a hardware failure and dynamic balancing of workloads. Individual ESXi hosts and clusters can exist at the same level under a datacenter object in vSphere.

**NOTE:** This lesson will use the vCenter server *(vcsa-01a.corp.local)* in your lab which has the Datacenter *(RegionA01)* and Cluster *(RegionA01-COMP01)* objects pre-existing.

List Available Clusters

In order to get a list of the clusters from the vCenter server that we are already connected to, we will use the "Get-Cluster" command.

1. Type the following command at the command line and press ENTER.

   Get-Cluster

2. We see it returns a list of all the clusters that we have on the *vcsa-01a.corp.local* vcenter server to which we were already connected.
Create a New Cluster

Creating a cluster on vSphere can be done with the "\texttt{New-Cluster}" cmdlet. A cluster can exist without any hosts in it. We are going to create a new cluster called "\texttt{MyCluster}" and set advanced properties on the cluster. When creating a cluster, it only needs a location and a name. Everything beyond these parameters is optional, but you can enable features that most use such as \textit{Distributed Resource Scheduler (DRS)} and \textit{High Availability (HA)}.

1. Type the following command at the command line and press \texttt{ENTER}.

   \begin{verbatim}
   New-Cluster -Location RegionA01 -Name MyCluster -HAEnabled -DRSEnabled
   \end{verbatim}

2. We see the script created a new cluster called "\texttt{MyCluster}" and has HA and DRS enabled for the advanced properties.
NOTE: If you are interested in other possible advanced properties that can be set when creating a new cluster, we can perform a "Get-Help New-Cluster". We will then see a number of additional parameters that can be defined.

Confirm the New Cluster Was Created.
Now that we have created the cluster, let’s confirm if the cluster was created. To verify the cluster was created, we need to run the "Get-Cluster" cmdlet.

1. Type the following command at the command line and press **ENTER**.

   ```powershell
   Get-Cluster -Name MyCluster
   ```

2. We see that the "MyCluster" is showing up proving that our previous commands worked.

## Remove a Cluster

Now we will remove the new cluster that we just created using the **Remove-Cluster** command since it was just to show how to create a new cluster. In the following steps, we will change the advanced properties on the existing **RegionA01-COMP01** cluster.

1. Type the following command at the command line and press **ENTER**.

   ```powershell
   Remove-Cluster -Cluster MyCluster -Confirm:$false -Verbose
   ```

2. Because we used the **-Verbose** switch, we can see the successful execution of the commands.
Now that we have created the cluster, let's confirm if the cluster was created. To verify the cluster was created, we need to run the "Get-Cluster" cmdlet.

1. Type the following command at the command line and press ENTER.

   Get-Cluster

2. We see that the "RegionA01-COMP01" is the only cluster showing which demonstrates successful execution.
Enable HA on the Cluster

The remainder of the steps in this lesson, we are going to make changes to the existing cluster RegionA01-COMP01 utilizing the Set-Cluster command. First we will enable High Availability (HA) for the RegionA01-COMP01 cluster and also use the "-Confirm:$false" command to disable asking if we are sure we want to enable it. This way we don't get the pop-up window asking.

1. Type the following command at the command line and press ENTER.

   ```
   Set-Cluster -Cluster RegionA01-COMP01 -HAEnabled:$true -Confirm:$false
   ```

2. We now see that HA has been enabled and reflects the status of "True" confirming the command worked properly.
Set HA Restart Priority to Low

The **HA Restart Priorities** settings specify the cluster HA restart priority. The valid values are **Disabled**, **Low**, **Medium**, and **High**. VMware’s HA is a feature that detects failed virtual machines and automatically restarts them on alternative vSphere hosts.

1. Type the following command at the command line and press **ENTER**.

   ```powershell
   Set-Cluster -Cluster RegionA01-COMP01 -HARestartPriority Low -Confirm:$false -Verbose
   ```

2. Because we used the **-Verbose** switch, we can see the execution information and confirmation that HA Restart Priority is now set to **Low**.

Set DRS Automation Level Settings - FullyAutomated

```powershell
Set-Cluster -Cluster RegionA01-COMP01 -DrsAutomationLevel FullyAutomated -Confirm:$false
```
Now we will change the DRS Automation Level back to "FullyAutomated" so that it will automatically vMotion virtual machines between hosts for us in the event the cluster becomes unbalanced.

1. Type the following command at the command line and press ENTER.

   ```
   Set-Cluster -Cluster RegionA01-COMP01 -DrsAutomationLevel FullyAutomated -Confirm:$false
   ```

2. We now see it shows that we have set the DrsAutomationLevel back to FullyAutomated on the cluster.

### Disable DRS on a Cluster

Now we will disable Distributed Resource Scheduler (DRS) for the RegionA01-COMP01 cluster using the "-DrsEnabled:$false" command.

1. Type the following command at the command line and press ENTER.

   ```
   Set-Cluster -Cluster RegionA01-COMP01 -DrsEnabled:$false -Confirm:$false
   ```

2. We now see that DRS has been disabled and reflects the status of "False" confirming the command worked properly.
Manage Resource Pools with VMware PowerCLI

Resource pools allow you to group together VM objects with similar performance requirements. They also allow a priority to be set on pools of compute, memory, and disk resources, so that when resource contention occurs, the hypervisor can decide which VMs get access to resources first.

For the purpose of this lesson, we’re going to configure two resource pools: Production and Development. The Production resource pool will be configured with the high setting for CPU and memory resources. The Development resource pool will be configured with the low setting for CPU and memory resources.

Report Resource Pool Details

We will start by locating the root Resources folder so that you can use it in the creation of a new resource pool.

1. Type the following command at the command line and press ENTER.

   ```ps
   Get-ResourcePool -Name Resources -Location RegionA01-COMP01
   ```

2. We see that it returns the resource pool for the entire cluster (or at the root level) in the lower pane.
Create Resource Pools

To create a new resource pool, we will use the **New-ResourcePool** cmdlet. In addition to the location, we also need to specify a name for the pool and additional parameters to define the CPU and RAM share settings. Additional parameters can also be defined to set reservations for CPU or RAM and expandable reservations.

1. Before we create a Resource Pool, we need to ensure that DRS is enabled on our cluster. Type the following command at the command line and press **ENTER**.

   ```powershell
   Set-Cluster -Cluster RegionA01-COMP01 -DrsEnabled:$true -Confirm:$false
   ```

2. We will find the location by identifying the root resource pool, **Resources** in our cluster, **MyCluster** and store it in a variable called **$myClusterRootRP**. Type the following command at the command line and press **ENTER**.

   ```powershell
   $myClusterRootRP = Get-Cluster RegionA01-COMP01 | Get-ResourcePool -Name Resources
   ```

3. We will use create our new resource pool, **Production**, in the location stored in **$myClusterRootRP**. Because this is our production resource pool, we will set both CPU and Memory Shares to High.

   ```powershell
   New-ResourcePool -Name Production -Location $myClusterRootRP -CPUSharesLevel High -MemSharesLevel High
   ```

4. We see that the resource pool has been created with the specified parameters.
Move VMs to the Resource Pool

Once the Resource Pool is created, we have to move objects into the Production resource pool. Moving a VM into the resource pool will not move it out of the folders or other locations where it might be assigned, it will only move the VM in the context of the Host and Clusters view. We will use the `Move-VM` cmdlet and specify the target resource pool using the `-Destination` switch.

1. Type the following command at the command line and press **ENTER**.

   ```
   Move-VM app-01a -Destination Production
   ```

Set Resource Pool Settings

1. Use the `Set-ResourcePool` cmdlet to set the Resource Pool Settings.

   ```
   Set-ResourcePool -ResourcePool Production -CpuShares level Normal
   ```
If you have an existing resource pool, we can adjust its settings using the `Set-ResourcePool` cmdlet.

1. Type the following command at the command line and press **ENTER**.

   ```powershell
   Set-ResourcePool -ResourcePool Production -CpuSharesLevel Normal
   ```

2. We see that the resource pool is now set to a **CpuSharesLevel** of **Normal**.
Create and Manage Folders with VMware PowerCLI

vSphere folders are containers which basically consists of other vSphere objects inside of them. Folders are not tied to physical resources. This means that VM objects from different clusters or even different datacenters can be logically grouped together. The same applies to port groups, switches, or datastores.

This is important as you begin to delegate access to VMs to service desk, developers, and other users in the organization, so that you can group together all of the VMs that a user needs to access. Folders also help administrators to easily locate objects and report on objects for a particular business unit or group within their companies.

In this lesson, you will look at the simple cmdlets used to create folder structures in vSphere, and move objects into these folders with simple VMware PowerCLI cmdlets.

NOTE: Whenever we name something that has a space in the name, we MUST enclose it in quotation marks ("Domain Controllers") around the entire name. If we don’t do that, VMware PowerCLI will see the "space" between the words and think that the word Controllers is supposed to be an argument. The script will then return an error since it is not an argument or proper syntax for PowerCLI.

List Existing Folders

Before we create a folder, let’s take a look at what folders are already created. We will use the Get-Folder cmdlet to retrieve the current list of folders.
NOTE: By running the "Get-Folder" command without any additional parameters, it will default to the datacenter and cluster that we are currently in at the moment. If we wanted to get the list of folders in a different datastore or cluster, we can do that by adding the additional parameters to the command. See the below example.

**Example:** Get-Datacenter -Name RegionA01 | Get-Folder

1. Type the following command at the command line and press ENTER.

   Get-Folder

2. We now see the list of all the existing folders.

### Create Folders Under the VM, Network, Host, and Datastore Folders

In the next couple of steps we will use the **New-Folder** cmdlet and see the different types of folders that it can create. Folders are used in vCenter for organizational purposes.

For this example, we will create several folder structures. We will create two, two-level folders under the VM and Templates view for Infrastructure and App Servers. We will create subfolders called "**Domain Controllers**" under Infrastructure. We will also create a "**Standard vSwitches**" folder in the Networks view and a "**NFS**" and "**iSCSI**" folder under the Datastores view. Finally, we will create a "**Engineering**" and "**ITOps**" folder under the Host and Clusters view to store clusters owned by these businesses.

### Create VM Folder

[Image of PowerShell command output showing creation of a new folder named "Infrastructure" under the VM and Templates view.]
Using **New-Folder**, you will pass in the desired name and the location parameter from the previous step to create a new folder. For this example, we use the `-Location` switch to specify both the datacenter, **RegionA01**, the root folder **VM**. Our new **Infrastructure** folder will inherit it's type from it's root folder.

1. Type the following command into the upper text pane.

```powershell
New-Folder -Name Infrastructure -Location (Get-Datacenter -Name RegionA01 | Get-Folder -Name VM)
```

2. We now see the new folder "**Infrastructure**" was created as a VM folder.

**Create New Folder Under the Infrastructure Folder**

The next step is to create a sub-folder under the Infrastructure folder we just created for our Domain Controllers. To do this, you can change the name and the location of the same cmdlet. Instead of searching for the folder named VM, you will search for the one we just created named Infrastructure.

**NOTE:** Whenever we name something that has a space in the name, we **MUST** use quotation marks ("Domain Controllers") around the entire name. If we don't do that, PowerCLI will see the "**space**" between the words and think that the word Controllers is supposed to be an argument. The script will then return an error since it is not an argument or proper syntax for PowerCLI.

1. Type the following command at the command line and press **ENTER**.

```powershell
Get-Folder -Name Infrastructure | New-Folder -Name "Domain Controllers"
```
2. We now see the new folder "Domain Controllers" was created.

List Existing Folders

Before we create a folder, let's take a look at what folders are already created. We will use the Get-Folder cmdlet to retrieve the current list of folders.

**NOTE:** By running the Get-Folder command without any additional parameters, it will default to the datacenter and cluster that we are currently in at the moment. If we wanted to get the list of folders in a different datastore or cluster, we can do that by adding the additional parameters to the command. See the below example.

1. Type the following command at the command line and press **ENTER**.

   ```powershell
   Get-Folder
   ```

2. We now see the list of all the existing folders including the two new folders which we created, **Infrastructure** and **Domain Controllers** folder.
Move VM to Infrastructure Folder

We will now move the `app-01a.corp.local` virtual machine into the **Infrastructure** VM folder using the **Move-VM** cmdlet.

1. Type the following command at the command line and press **ENTER**.

   ``` Powershell
   Move-VM -VM app-01a -InventoryLocation Infrastructure
   ```

Creating Other Folder Types

Just like we did with creating VM folders, we can also create other folder types to be more organized in the vCenter server. Below are some examples of using the **New-Folder** command to create new folders for Datacenters, Datastores, and vSwitches. You can then also use the "**Move-**" commands to move Datastores, Datacenters, and Clusters under the folders you create.

**NOTE**: In the below examples, we use the actual names of the Datacenter, Datastore, and Cluster from the lab environment to make it a little more real world examples.

**New & Move Datacenter Folder Examples:**

```
Get-Folder -NoRecursion | New-Folder -Name MyDatacenter
Move-Datacenter -Datacenter RegionA01 -Destination MyDatacenter
```

**New & Move Datastore Folder Examples:**
New-Folder -Name InfrastructureDS -Location (Get-Datacenter -Name RegionA01 | Get-Folder -Name datastore)

Move-Datastore -Datastore RegionA01-ISCSI01-COMP01 -Destination InfrastructureDS

**New & Move Host & Cluster Folder: Examples**

New-Folder -Name InfrastructureH -Location (Get-Datacenter -Name RegionA01 | Get-Folder -Name host)

Move-Cluster -Cluster RegionA01-COMP01 -Destination InfrastructureH
Conclusion

Congratulations on completing Module 2!

In this module, we learned how to Automate configuration activities for the vCenter server using VMware PowerCLI.

The Module contained the following lessons:

- **Getting Started With PowerShell and PowerCLI:** We learned how to launch PowerCLI and open a few panes to make it easier to work with.
- **Connect to the vCenter Server Using VMware PowerCLI:** We learned how to connect to a vCenter instance using VMware PowerCLI, this is the first step which needs to be done to work with VMware PowerCLI.
- **Create a Datacenter in vCenter Using VMware PowerCLI in Windows:** We learned how to create a datacenter using Windows PowerShell and VMware PowerCLI.
- **Connect to vCenter and Create a Datacenter Using VMware PowerCLI in Linux:** We learned how to create a datacenter using PowerShell core on Linux and VMware PowerCLI.
- **Get and Set Cluster Settings:** We learned how create a cluster, set advanced properties for the cluster, and also get the properties using VMware PowerCLI.
- **Manage Resource Pools with VMware PowerCLI:** We learned how to manage Resource Pools using VMware PowerCLI.
- **Create and Manage Folders with VMware PowerCLI:** We learned how to create and manage folders using VMware PowerCLI.

**VMware PowerCLI Resource Links:**

If you did not start this lab at the beginning of Module 1 and would like to see a list of resource links for VMware's PowerCLI, please click on the below link which will take you to Module 1's Conclusion which contains the links.

- **Module 1 - Conclusion**

**OPTIONAL: How to End the Lab**

You can now continue to the next module by clicking forward, or use the Table of Contents to skip to another desired Module.
If you don’t plan to take any more modules in the lab at this time, click on the END button.

**Note:** If you end your lab, you will need to re-register for the lab in order to take any other modules.
Module 3 - Automate Configuration Activities for ESXI Hosts (15 minutes)
Introduction

In this module, you will learn how to Automate configuration activities for ESXi hosts using PowerCLI.

This Module contains the following lessons:

- **Connect to vCenter Server Using VMware PowerCLI in Linux**: We learn how to connect to a vCenter instance using PowerCLI and bypass the invalid certificate error which can come the first time you connect to a vCenter instance.
- **List ESXi Hosts Within the vCenter Server**: We learn how to report all the available objects with ESXi host also report the methods and parameters available.
- **Adding Hosts and Managing Status**: We learn how to access and work with the ESX objects using PowerCLI.
- **Get Datatstore Information and Rescan HBAs and VMFS Datastores**: We learn how to interact with datastores using PowerCLI.
Connect to vCenter Using VMware PowerCLI in Linux

This lesson helps define the steps which needs to be performed to connect to a vCenter server. To execute any cmdlet in VMware PowerCLI, you need to connect to the vCenter using the **Connect-VIServer** cmdlet.

By default, VMware PowerCLI cmdlets run on the vCenter Server systems or vCloud Director (vCD) servers you are already connected to or whichever is the default. When you connect to a vCenter Server system by using the **Connect-VIServer** cmdlet, the server connection is stored in the `$DefaultVIServers` array variable. This variable contains all connected servers for the current VMware PowerCLI session. To remove a server from the `$DefaultVIServers` variable, you can either use **Disconnect-VIServer** to close all active connections to this server, or modify the value of `$DefaultVIServers` manually.

When you connect to a vCloud Director system by using **Connect-CIServer**, the server connection is stored in the `$DefaultCIServers` array variable. This variable contains all connected servers for the current session. To remove a server from the `$DefaultCIServers` variable, you can use **Disconnect-CIServer** cmdlet.

For this module, we will be using PowerShell core on Linux. Windows PowerShell with VMware PowerCLI will us the same commands therefore, if it is already open and you prefer to use that, you can skip this lesson and go to the next lesson.

**SSH to a Linux VM**

To begin using PowerCLI, we must start SSH into a Linux VM with **PowerShell** and **PowerCLI** already installed. As discussed in previous modules, we have prepared a virtual machine, `app-01a.corp.local` with both PowerShell and PowerCLI already installed.
1. Double-click on the **Putty** icon on the desktop.

2. Enter **app-01a.corp.local** in the Host Name (or IP address) box

3. Click Open

4. If a security alert appears, click **Yes** to add the key to PuTTY's cache and continue.
4. Add the server's host key to the registry by clicking **Yes**.

5. Enter the username **hol**
6. Enter the password **VMware1!**
Start PowerShell

1. Type in the following command and press **Enter**

```
pwsh
```

Set PowerCLI Configuration to Ignore Invalid Certificates
1. Type in the following command and press **Enter**

   ```plaintext
   Set-PowerCLIConfiguration -InvalidCertificateAction Ignore
   ```

2. Confirm the action for all by typing **a** and **Enter**

**Connect to the vCenter Server**

Next, we must connect to our vCenter server. You need to use the **Connect-VIServer** cmdlet.

1. Type in the following command at the command line and press **ENTER**.

   ```plaintext
   Connect-VIServer 192.168.210.22 -User administrator@corp.local -Password VMware1!
   ```

Due to lab constraints, this command may not succeed the first time. If you receive an error, please execute the command a second time.

2. When you have successfully logged into the vCenter server, a prompt will be displayed that shows the server you are connected to, using Port 443, and the user account in the output pane.
List ESXi Hosts Within the vCenter Server

In this lesson, we will explore the VMware PowerShell objects emitted by the ESXi Host in VMware PowerCLI as well as look at some methods and parameters. We will use the "Get-VMHost" command to get information about what hosts are available in a vCenter server, datacenter, cluster, and on what host a specific virtual machine is running.

List ESXi Hosts

Cmdlets become available to manage a host after we connect to that host to manage it. The first concept that you will need to become aware of are VMware PowerShell objects. Objects are defined as data obtained from commands that run in Windows PowerShell and VMware PowerCLI. To perform configuration on an ESXi host, the commands that we run will need a host object, which is specified. To retrieve a list of all host objects that are in the vCenter server that we are connected to already, we will need to use the "Get-VMHost" command.

1. Type the following command at the command line and press ENTER.

   Get-VMHost

2. We see that the command returned a list with two ESXI hosts which are the hosts connected to the vcsa-01b.corp.local vCenter server.
List ESXi Hosts from a Specific Datacenter

We will now add the "-Location" onto the "Get-VMHost" command. By adding this, we can get the ESXi hosts that belong to a specific vCenter server, datacenter, or cluster. We are currently connected to the **vcsa-01b.corp.local** vCenter server which has only the one datacenter (**RegionB01**).

1. Type the following command at the command line and press ENTER.

   ```bash
   Get-VMHost -Location RegionB01
   ```

2. We see that the command returned the same two ESXi hosts since we pointed to the same datacenter **RegionA01** as the previous step. This is because our vCenter Server has only one Datacenter.

List ESXi Host Where a Specific VM is Located

There may be a time where you don't know which ESXi host you want to connect to because you want to connect to a host that a specific virtual machine is running on. To figure out what host a virtual machine is on, we will use the "Get-VMHost -VM vm-name" command. In this example, we want to know what host the "app-01b.corp.local" virtual machine is on.

1. Type the following command at the command line and press ENTER.

   ```bash
   Get-VMHost -VM app-01b
   ```

2. We see that the command returned the host **"esx-01b.corp.local"** because that is where the "**app-01b" virtual machine is currently running.
Adding Hosts and Managing Status

In this lesson, we will discuss how to add and remove a host from a vCenter server using VMware PowerCLI commands.

Adding a Host

Using VMware PowerCLI, we can add a new host to a datacenter in a vCenter server. Below is an example of the commands we would use to get the list of hosts and then add a specific host. In this step, we will add a host, `esx-03b.corp.local`, to the `RegionB01-COMP01` cluster. The default username for a host is always "root" and then we are using the password of `VMware1!`.

1. Type the following command at the command line and press ENTER.

   ```
   Add-VMHost -Name esx-03b.corp.local -Location (Get-Cluster RegionB01-COMP01) -User root -Password VMware1! -Force
   ```

2. We can see from the command response that our host, `esx-03b.corp.local` has been successfully added to the specified cluster.

   NOTE: If we wanted to join the new host to the datacenter but not to an existing cluster, we replace "Get-Cluster RegionB01-COMP01" with "Get-Datacenter Region01".
Put a Host in Maintenance Mode

1. Type the following command at the command line and press **ENTER**.

   ```powershell
   Set-VMHost -VMHost esx-03b.corp.local -State "Maintenance" -RunAsync
   ```

2. To verify the status of the host, type the following command at the command line and press **ENTER**.

   ```powershell
   Get-VMHost
   ```

3. From the output of this command, we can validate that the host **esx-03b.corp.local** is now in Maintenance mode.

   If the host is not automated or is partially automated and has powered-on virtual machines running on it, you must use the RunAsync parameter and wait until all powered-on virtual machines are relocated or powered off.
**Take a Host out of Maintenance Mode**

1. Type the following command at the command line and press **ENTER**.

   ```powershell
   Set-VMHost -VMHost esx-03b.corp.local -State "Connected"
   ```

2. To verify the status of the host, type the following command at the command line and press **ENTER**.

   ```powershell
   Get-VMHost
   ```

3. From the output of this command, we can validate that the host **esx-03b.corp.local** is now in **Connected**.
Get Datastore Information and Rescan HBAs and VMFS Datastores

In this lesson, we will use VMware PowerCLI to get information on our datastores from our vCenter server. We will also do some commands to rescan our Virtual Machine File System (VMFS) datastores and Host Bus Adapters (HBA).

Get a List of Datastores

In this lesson we will use the "Get-Datastore" command to retrieve a list of all datastores from the vCenter server we are currently connected to.

1. Type the following command at the command line and press ENTER.

   Get-Datastore

2. We see it returns the datastores "RegionB01-ISCSI01-COMP01" that is available in the vCenter server to which we are connected.
Get Information of a Specific Datastore

In this lesson we will use the "Get-Datastore" command to retrieve a list of all datastores from the vCenter server we are currently connected to.

1. Type the following command at the command line and press ENTER.

   Get-Datastore -Name RegionB01-ISCSI01-COMP01

2. We see it returns the same datastore information as running the Get-Datastore command we ran previously. That is because it is the only datastore that we have in this vCenter server. If we had multiple datastores, then using the "-Name" attribute would be more useful to return the information associated only to the named datastore.
Rescan All HBAs

Sometimes as a VMware administrator and working with datastores, we make changes to the host settings or the storage itself. In these cases, we need to sometimes rescan the **Host Bus Adapters (HBA)** on the hosts for the settings to show up. In order to do this, we will use the "Get-VMHostStorage" command along with the "-RescanHba" attribute. We will run the command against the "esx-01b.corp.local" host in our environment.

1. Type the following command at the command line and press **ENTER**.

   ```powershell
   Get-VMHostStorage -VMHost esx-01b.corp.local -RescanAllHba
   ```

2. We see it does not return anything regarding the rescanning of the HBAs. It only reflects that **SoftwareIScsi** is enabled. However, since it is just a rescan, there are no setting changes that are made, so this is an expected output from running the rescan command.
Rescan VMFS Datastores

In the event we need to do a rescan for new or updated Virtual Machine File System (VMFS) datastores, we can use the "-RescanVmfs" attribute.

1. Type the following command at the command line and press ENTER.

   `Get-VMHostStorage -VMHost esx-01b.corp.local -RescanVmfs`

2. We see it does not return anything regarding the rescanning of the VMFS datastores just like when we did a rescan for all HBAs.
Conclusion

Congratulations on completing Module 3!

In this module, we learned how to Automate configuration activities for ESXi hosts using VMware PowerCLI. Since hosts are within clusters and datacenters, we also covered some VMware PowerCLI commands against them as well as the hosts themselves.

The Module contained the following lessons:

- Connect to vCenter Server Using PowerCLI in Linux: We learn how to connect to a vCenter instance using PowerShell Core for Linux and PowerCLI.
- List ESXi Hosts Within the vCenter Server: We learn how to report all the available objects with ESXi host also report the methods and parameters available.
- Adding Hosts and Managing Status: We learn how to access and work with the ESX objects using PowerCLI.
- Get Datastore Information and Rescan HBAs and VMFS Datastores: We learn how to interact with datastores using PowerCLI.

VMware PowerCLI Resource Links:

If you did not start this lab at the beginning of Module 1 and would like to see a list of resource links for VMware's PowerCLI, please click on the below link which will take you to Module 1's Conclusion which contains the links.

- Module 1 - Conclusion

OPTIONAL: How to End the Lab

You can now continue to the next module by clicking forward, or use the Table of Contents to skip to another desired Module.

If you don't plan to take any more modules in the lab at this time, click on the END button.

Note: If you end your lab, you will need to re-register for the lab in order to take any other modules.
Module 4 - Automate Virtual Machine Management (30 minutes)
Introduction

In this module, you will learn how to Automate configuration activities for the vCenter server using VMware's PowerCLI command-line tool.

This Module contains the following lessons:

• **Connect to the vCenter Server Using VMware PowerCLI:** We learn how to connect to a vCenter server using VMware PowerCLI, this is the first step which needs to be done to start creating or modifying vCenter server objects.

• **Working with VM Templates:** We learn how to create a template and clone it to create a new virtual machine.

• **Managing Virtual Machine Configuration:** We learn how to use the Set-VM cmdlet to reconfigure a virtual machine.

• **Managing Hard Disk Configuration:** We learn how to use the Set-HardDisk cmdlet to configure virtual machine storage.

• **Managing Virtual Machine Power States:** We learn how restart, shut-down and power-on virtual machines.

• **Managing VM Snapshots:** We learn how to create and manage virtual machine snapshots.

Introducing VMware PowerCLI

VMware PowerCLI offers an excellent command-line alternative to the point-and-click based administration of the vSphere Web Clients. As virtualization has become mainstream and deployments begin to sprawl, the simple commands of VMware PowerCLI allows faster administration by executing tasks against large groups of objects in the virtual environment.

VMware PowerCLI follows a very logical pattern, it can be quickly adopted and understood, making it the first choice for many vSphere administrators. It also combines extensibility to allow users to build their own functions and modules to solve specific problems not addressed by out-of-box functionalities present in the vCenter Graphical User Interface (GUI) console.

The focus of this lab session will be on VMware PowerCLI in the context of managing the virtual machines.

Overview/Benefits

VMware helps enterprise IT overcome cloud management challenges with the following differentiators:
• **Achieving Fastest Time-to-Cloud Value:** extensive out-of-the-box capabilities, extensible architecture and ability to leverage existing investments, enables you to create scalable private, public and desktop cloud services in just days

• **Rapid Ecosystem Integration:** a full spectrum of extensibility options that empowers IT personnel to enable, adapt, and extend their cloud to work within their existing IT infrastructure and processes, thereby eliminating expensive service engagements while reducing risk

• **Embracing Consumerization of IT:** unifies disparate systems and processes into a highly personalized, self-service experience governed by user and organizational awareness, ensuring the exact needs of the user are met in the context of the business

• **Boosting IT Efficiency:** VMware's advanced self-service and automation capabilities lead to a significant reduction in Operational Expenditures (OpEx), while organizationally aware governance enables a multi-tenant infrastructure, improving hardware utilization and eliminating Capital Expenditures (CapEx)

• **Increasing Business Agility:** enterprise IT is empowered to quickly enable cloud services so that the business can react quickly to changing market demands and capture market share

• **Hybrid Cloud Automation:** agnostic provisioning of hardware platforms, operating systems, hypervisors, management tools, and public clouds; vRealize Automation allows customers to rapidly stand up cloud services to deliver quick time to value from your IT investments
Connect to the vCenter Server Using VMware PowerCLI

This lesson helps define the steps which needs to be performed to connect to a vCenter server. To execute any cmdlet in VMware PowerCLI, you need to connect to the vCenter using the `Connect-VIServer` cmdlet.

By default, VMware PowerCLI cmdlets run on the vCenter Server systems to which you are already connected. When you connect to a vCenter Server system by using the `Connect-VIServer` cmdlet, the server connection is stored in the `$DefaultVIServers` array variable. This variable contains all connected servers for the current VMware PowerCLI session. To remove a server from the `$DefaultVIServers` variable, you can either use `Disconnect-VIServer` to close all active connections to this server, or modify the value of `$DefaultVIServers` manually.

Start PowerShell

To begin using PowerCLI, we must start **Windows PowerShell**. As discussed in the previous module, PowerCLI has already been installed.

1. Double-click on the **Windows PowerShell** icon on the desktop.
Next, we must connect to our vCenter server. You need to use the **Connect-VIServer** cmdlet.

1. **Type in the following command at the command line and press ENTER.**

   ```powershell
   Connect-VIServer vcsa-01a.corp.local -User administrator@corp.local -Password VMware1!
   ```

2. **When you have successfully logged into the vCenter server, a prompt will be displayed that shows the server you are connected to, using Port 443, and the user account in the output pane.**
Working With VM Templates

Cloning is one of the most sought after features of virtualization. Once you use this feature, you will never want to go back to manually deploying VMs. It really eases the problems of deploying consistent and compliant virtual machines by allowing an administrator to create a good, compliant image and then make copies of it for all future deployments.

While you can clone regular virtual machines, most of the time you will be cloning new virtual machines from a template. In this lesson, you will explore how to clone a VM and how to convert it to a template.

The clone functionality is part of the **New-VM** cmdlet

**List Existing Virtual Machines**

VMware PowerCLI has cmdlets which allow you to search for existing virtual machines. To start, let's run the **Get-VM** cmdlet with no additional parameters to see what VM's exist in vCenter.

1. Type the following command at the command line and press **ENTER**.

   ```
   Get-VM
   ```

2. You will see that the command returns the name of four existing VM's including template-01a. While this VM is named template-01a, it is not a template. Therefore, our next step will be to convert template-01a from a virtual machine to a template.
Convert Clone VM to template

We will use the Set-VM to change the properties of our template-01a virtual machine. The -ToTemplate switch will convert the virtual machine to a template.

1. Type the following command at the command line and press ENTER.

   Set-VM -VM template-01a -ToTemplate -Confirm:$false

2. You will see that the command returns the name of virtual machine acted upon, template-01a, as an indication that the command was completed successfully.

Clone a New Virtual Machine from a Template

Clone a New Virtual Machine from a Template
We will use the New-VM to provision a new virtual machine by cloning template-01a. This new virtual machine, HOL-VM1, will be placed on esx-01a.corp.local and will use thin provisioning.

1. Type the following command at the command line and press **ENTER**.

   ```powershell
   New-VM -Template template-01a -Name HOL-VM1 -VMHost esx-01a.corp.local -DiskStorageFormat Thin
   ```

2. You will see that the command returns the name of our new virtual machine, HOL-VM1, as an indication that the command was completed successfully.
Managing Virtual Machine Configuration

The Set-VM commandlet allows us to manage the virtual machine configuration. If the OSCustomizationSpec parameter is used, the cmdlet customizes the virtual machine according to the specification. In addition, the cmdlet allows you to revert a virtual machine to a snapshot and convert a virtual machine to a template.

Reconfigure Memory and CPU

In this step, we are going to reconfigure the virtual machine we created in the prior step. We are going to set the number of vCPU to 4 and the Memory to 8GB.

1. Type the following command at the command line and press ENTER.

   ```powershell
   Set-VM HOL-VM1 -NumCpu 4 -MemoryGB 8
   ```

2. At the confirmation prompt, enter a to confirm all the configuration changes and press ENTER.

3. You will see that the VM has been reconfigured with the desired configuration parameters.
Set-Restart Priority

In this step, we are going to change the restart priority for our virtual machine to High. The first thing that we need to do is ensure that HA is enabled on our cluster.

1. Type the following command at the command line and press **ENTER**.
   
   ```powershell
   Set-Cluster -Cluster RegionA01-COMP01 -HAEnabled:$true
   ```

2. At the confirmation prompt, enter **y** to confirm that we want to enable HA on this cluster and press **ENTER**.

3. Following reconfiguration of the cluster, we see a prompt indicating that HA has now been enabled.

4. Now we will set the restart priority for our virtual machine. Type the following command at the command line and press **ENTER**.
   
   ```powershell
   Set-VM HOL-VM1 -HARestartPriority High
   ```

5. While the output of this command doesn't show a restart priority, it is an indication that the command was successfully executed.
Managing Hard Disk Configuration

The Set-HardDisk cmdlet allows us to modify the properties of the specified virtual hard disk. You can change the size and the persistence type, and inflate or expand the specified virtual hard disk. Do not use the Inflate parameter at the same time with the Persistence and CapacityGB parameters. If a helper virtual machine is used, all virtual machines associated with the disk and the helper virtual machine should be powered off before expanding the disk. When you resize more than one disks using a helper virtual machine, the disks are resized one by one causing the helper machine to power on and off for each virtual machine and this might slow the cmdlet performance.

Expanding a Hard Disk

In this step, we are going to reconfigure the virtual machine we created in the prior step. We are going to set the number of vCPU to 4 and the Memory to 8GB.

1. Type the following command at the command line and press ENTER.

   \[Get-HardDisk HOL-VM1\]

2. You will see that the virtual machine as a single hard disk of 3 GB.

3. Now we will pipe that hard disk into the Set-HardDisk command to increase it to 10GB. Type the following command at the command line and press ENTER.

   \[Get-HardDisk HOL-VM1 | Set-HardDisk -CapacityGB 10\]
4. At the confirmation prompt, enter `a` to confirm all the configuration changes and press `ENTER`.
5. You will see that the Hard Disk has been reconfigured as desired.

If you have more than one disk on the VM, unless you want to grow all the disks to the same size, you will need to select one of them. You can do this by specifying parameters such as filename, path, or datastore path. A more advanced method is to use a **Where** clause on the Get-HardDisk cmdlet. The **Where** clause allows you to provide a conditional statement that you can use to specify which disk(s) you want to edit.
Managing Virtual Machine Power States

A series of cmdlets allows us to manage the power state of our virtual machines. In this lesson, we will examine a few of those cmdlets.

Rebooting a Virtual Machine or Guest Operating System

In this step, we are going to examine two methods of restarting a virtual machine.

1. First, we will do a soft restart by simply restarting the Operating System. Type the following command at the command line and press \texttt{ENTER}.

   \begin{verbatim}
   Restart-VMGuest app-01a -Confirm:$false
   \end{verbatim}

2. You will see that after the operating system has rebooted, we will get an indication of the Operating System Status, IP address and type.

3. Now we will do a hard reboot of the virtual machine. Type the following command at the command line and press \texttt{ENTER}.

   \begin{verbatim}
   Restart-VM app-01a -Confirm:$false
   \end{verbatim}

4. You should notice that this command will take a bit longer. We now get an indication of the virtual machine name and it's status.
In this step, we are going to examine two methods of powering down a virtual machine.

1. First, we will do a soft power-off by shutting down the Operating System. Type the following command at the command line and press ENTER.

   Stop-VMGuest kms-01a -Confirm:$false

2. From the response, we can see that the virtual machine is still running. Because this is a soft power-off, it will take a few minutes. We will revisit this after our next step.

3. Now we will do a hard power-off of the virtual machine. Type the following command at the command line and press ENTER.

   Stop-VM app-01a -Confirm:$false

4. From the response, we can see that the virtual machine was immediately powered-off.

5. Now we will execute a Get-VM to get a listing of the virtual machines and their power-state. If kms-01a shows a state of PoweredOn, wait another moment and re-run the Get-VM command. Type the following command at the command line and press ENTER.

   Get-VM
6. After the operating system has shut down, the virtual machine will enter a Powered-Off state.

### Powering on a Virtual Machine

![Powering on a Virtual Machine](image)

For our final step in this lesson, we will power-on the app-01a virtual machine.

1. First, we will do a soft restart by simply restarting the Operating System. Type the following command at the command line and press **ENTER**.

   Start-VM app-01a

2. After a moment, you will see a prompt indicating that the virtual machine is now Powered-On.
Managing VM Snapshots

Snapshots are one of the best features in vSphere. Snapshots are the safety net built into the platform that allows you to easily revert to a previous known good state in the event that something happens in a VM. Some environments run regularly scheduled snapshots. Some administrators use them only before changes are made in the environment.

There are two types of snapshots. You can take snapshots that include the memory to return the VM to a running state, including the memory at the time that it is taken. This type of snapshot allows a VM to be brought back to a running state with an active application. There are also snapshots that simply snap the disk, but these would make a crash-consistent version of the virtual machine since the running application might have data in memory that has not been preserved. Reverting to this type of snapshot would boot the virtual server as if it had been reset while running or like it had crashed.

Over a period of time, virtual machine snapshots grow to a point that they can exhaust all of the available disk space on a datastore. If kept for a long time or if a number of snapshots are taken, they can also effect performance of the VM. Administrators can take multiple snapshots and each snapshot references a parent, which means that there is added overhead for IO since the system has to combine multiple files in order to find the correct data to return.

In this chapter, you will start with simple cmdlets to create, report, and remove snapshots.

Creating Snapshots Using PowerCLI

To create a new snapshot, you will use the New-Snapshot cmdlet. It requires two parameters, -VM for the VM to be targeted and -Name for the name of the snapshot. The description can be any string of characters. It can be used to add notes about the snapshot, such as who requested it or a date when it can be safely removed.
1. Type the following command at the command line and press **ENTER**.

```powershell
New-Snapshot -VM app-01a -Name "app-01a basic" -Description "This is a basic snapshot"
```

2. Once the snapshot has been created, you will see feedback indicating that the command has been completed successfully. Note that the Power State indicates that the snapshot has been taken with no memory included as though the virtual machine was in a powered off state.

You can also create a snapshot that quiesces the disk. This means that it temporarily stops all writes so that it can take a snapshot of the disk knowing that there is no activity in progress. To do this, add the `-Quiesce` parameter.

1. Type the following command at the command line and press **ENTER**.

```powershell
New-Snapshot -VM app-01a -Name "app-01a quiesce" -Description "This is a quiesced snapshot" -Quiesce
```

2. Once the snapshot has been created, you will see feedback indicating that the command has been completed successfully.
Creating a snapshot with only the required parameters creates a powered off snapshot of the disk, which is crash-consistent. This means that any data in memory might not be preserved in the snapshot and the VM would think that it started from a crash state on the next boot. However, you can also include a snapshot on the RAM for the VM to revert to a powered on state. To do this, add the -Memory parameter.

1. Type the following command at the command line and press ENTER.

```
New-Snapshot -VM app-01a -Name "app-01a memory" -Description "This is a snapshot w/ memory" -Memory
```

2. Once the snapshot has been created, you will see feedback indicating that the command has been completed successfully. Because this snapshot included memory, it shows in the Powered On state.

### Removing Snapshots

First, let’s take a look at the snapshots we've created in the prior steps of this lesson.

1. Type the following command at the command line and press ENTER.

```
Get-Snapshot app-01a
```

2. The response from this command will show all three snapshots we took earlier.

3. Now, we will delete the snapshot called **app-01a basic**. Type the following command at the command line and press ENTER.
Get-Snapshot app-01a -Name "app-01a basic" | Remove-Snapshot

4. Because, this command has no response prompts, let's use the Get-Snapshot cmdlet to see what snapshots are still associated with app-01a. Type the following command at the command line and press **ENTER**.

```powershell
Get-Snapshot app-01a
```

5. The results of this cmdlet execution show us that the app-01a basic snapshot has been deleted.
Conclusion

Congratulations on completing Module 4!

In this module, we learned how to Automate configuration activities for Virtual Machines using VMware PowerCLI.

VMware PowerCLI Resource Links:

If you did not start this lab at the beginning of Module 1 and would like to see a list of resource links for VMware's PowerCLI, please click on the below link which will take you to Module 1's Conclusion which contains the links.

• Module 1 - Conclusion

OPTIONAL: How to End the Lab

You can now continue to other content in this lab by using the Table of Contents to skip to the desired section.

If you don’t plan to take any more modules in the lab at this time, click on the END button.

Note: If you end your lab, you will need to re-register for the lab in order to take any other modules.
Conclusion

Thank you for participating in the VMware Hands-on Labs. Be sure to visit http://hol.vmware.com/ to continue your lab experience online.

Lab SKU: HOL-1911-05-SDC

Version: 20190914-212200