# Table of Contents

Lab Overview - HOL-1811-6 - vSphere Automation API and SDK ........................................ 2
  Lab Guidance ....................................................................................................................... 3
Module 1 - API and DCLI (45 minutes) ............................................................................. 11
  Introduction ......................................................................................................................... 12
  Invoke vCenter REST API calls with Postman Client ..................................................... 14
  Exploring DCLI features and workflow .......................................................................... 29
Module 2 - Python and the vSphere API (45 minutes) ....................................................... 44
  Introduction ......................................................................................................................... 45
  Exploring the Python SDK Directory Structure ............................................................... 47
  Managing Virtual Machine using Python and vSphere Automation API ................... 51
Module 3 - Java and the vSphere API (45 minutes) ............................................................ 61
  Introduction ......................................................................................................................... 62
  Exploring the Java SDK directory structure .................................................................... 64
  Managing Virtual Machine using Java and vSphere Automation API ....................... 67
Module 4 - API Explorer (30 minutes) .............................................................................. 70
  Introduction ......................................................................................................................... 71
  Accessing the API Explorer ............................................................................................. 72
Lab Overview - HOL-1811-6 - vSphere Automation API and SDK
Lab Guidance

Note: It will 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.

This Lab introduces you to vSphere Automation APIs and SDKs, which are new developer friendly, simplified interfaces that allow for Virtual Machine creation, modification and deletion via a consistent set of developer and automation tooling.

The lab is broken into 3 Modules which can be taken in any order:

Module 1 - API and DCLI (45 minutes - Basic)

This module shows the simplicity and ease of use in managing VMs with the vSphere Automation API and Multi-Platform CLI.

Module 2 - Python and the vSphere API (45 minutes - Basic)

This module explores the tight integration between the new vSphere Automation SDK for Python and Pyvmomi to perform common vSphere tasks in Python.

Module 3 - Java and the vSphere API (45 minutes - Basic)

This module explores the integration of Java with the vSphere APIs to perform a set of vSphere tasks.

Lab Captains:

  • Kevin Steil, Global Solutions Consultant, (Modules 1-4)

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:

Introduction to the Scenario

Rainpole Systems is an electronics manufacturer located in Palo Alto, CA. Rainpole designs and manufactures electronic devices for use in everything from aircraft instrumentation to home automation. Given the diversity of their product set, Rainpole needs to develop cloud-based IT services to support the increasing demands of the business units.

Rainpole Systems in planning to use vCenter API's to automate the build, test and deployment of their custom developed eCommerce application. They also plan to leverage the Python and Java SDK's for vCenter API's for IT DevOps to minimize the chance of errors impacting their production cloud environment.

VMware vCenter REST API's simplifies the automation of complex IT tasks and integrates with REST clients like POSTMAN and also comes integrated with Java and Python via sdk's to adapt and extend service delivery and operational management, effectively working with existing infrastructure, tools and processes. You have the most knowledge around existing processes and have been tasked with coming up to speed on VMware vCenter REST API's and starting down the path of automating existing processes.

As you work through the lab's use cases, you'll be assuming roles within a fictional company, Rainpole, and addressing their business and IT challenges.

You will take on the roles of a Rainpole Cloud Administrator, a Rainpole Developer and a Development Manager in this exciting lab exercise to experience how VMware vCenter API's can help make these goals a reality.
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 cannot 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.

Control Center

You should see the ControlCenter Desktop when you first log into the Lab.
At any point during this lab you can verify that you are on the Control Center Desktop by examining the background wallpaper.

**Applications Used For This Lab**

We will be using the POSTMAN client for the REST workflows and the windows command prompt for the JAVA and SDK Part.

**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 your lab has not changed to "Ready", please ask for assistance.
Module 1 - API and DCLI
(45 minutes)
Introduction

In this module, you will learn how to build and invoke vCenter REST API calls using Postman Client and DCLI.

If you have not already reviewed it, the Lab Introduction contains details about Rainpole Systems (our example company), as well as important information on text entry, multiple language support and on-screen keyboard configuration. If desired, you can click here to review this information again.

This Module contains the following lessons:

- **Invoke vCenter REST API calls with Postman Client**: Learn how to build and invoke vCenter REST API calls using the Postman client.
- **Exploring DCLI features and workflow**: Learn how to automate various vCenter administrative tasks using Datacenter Command Line Interface (DCLI).

Introducing VMware vCenter REST API

vCenter REST API is a new developer friendly, simplified API which allows for Virtual Machine creation, modification and deletion via a consistent set of developer and automation tooling. This will allow for consistent interaction with the API via the tools which make sense to the developer community or someone automating VMware products.

The focus of this lab session will be on vCenter REST API Calls.

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
Invoke vCenter REST API calls with Postman Client

vCenter REST API is a new developer friendly, simplified API which allows for Virtual Machine creation, modification and deletion via a consistent set of developer and automation tooling. This will allow for consistent interaction with the API via the tools which make sense to the developer community or someone automating VMware products, included in this are:

- Multiple SDKs
- PowerCLI
- Datacenter CLI (DCLI)
- vRealize Orchestrator

This new feature of this API allow use of:

- **Cluster** - Get/List
- **Datacenter** - Create/Get/List/Delete
- **Datastore** Get/List
- **Folder** - List
- **Host** - Create/Get/List/Delete/Connect/Disconnect
- **Network** List
- **Resource pool** Get/List
- **VM** - Create/Get/List/Delete
  - **Power** Get/Start/Stop/Suspend/Reset
  - **Hardware** Get/Update/Upgrade
    - **Boot** Get/Update
      - **Device** Get/Set
      - **Disk** Get/Set
  - **Cdrom** - Create/Get/List/Delete/Update/Connect/Disconnect
  - **Cpu** Get/Update
  - **Disk** - Create/Get/List/Delete/Update
  - **Ethernet** - Create/Get/List/Delete/Update/Connect/Disconnect
  - **Floppy** - Create/Get/List/Delete/Update/Connect/Disconnect
  - **Memory** Get/Update
  - **Parallel** - Create/Get/List/Delete/Update/Connect/Disconnect
  - **Serial** - Create/Get/List/Delete/Update/Connect/Disconnect
  - **Adapter**
    - **Sata** Create/Get/List/Delete
    - **SCSI** - Create/Get/List/Delete/Update

In previous versions of vSphere the primary API for management has been SOAP based and used heavily by partners through SDKs and by Administrators via vCLI, PowerCLI or vRealize Orchestrator.
In today’s modern development world the API, Developers and Automation have become a focus of modern IT, the new vCenter REST API has been designed to fulfill the needs of a modern API by providing a simple, modern and expansive API surface and consistent tools and SDKs to access this.

During the workflows on the next pages you will realize that even if you are not a developer you will be able to make use of the API and tooling to automate some of your Virtual Machine tasks.

**Launching the Postman Client**

![Postman UI](image)

Postman is a free application which can be used to organize and send requests to a REST based API, you will be using this during the steps below to work with this new API.

1. Launch PostMan by clicking the 'PostMan' icon present on the Desktop

**Become familiar with Postman UI to call the vCenter API**

Next you will explore the contents of each section of the Postman client

The below diagram explains what each section in Postman will be used for.

1. Gives us a list of pre-created REST actions we can use to send to the API. This section also shows a set of sample actions has been setup to show the vCenter REST API which takes you through a common workflow and shows how simple and consistent it is to call the API and take a VM through its lifecycle. Postman has been configured with a variable in the place of the vCenter name, you will see {{vc}} which will translate to the IP/Hostname of the VC we will use in the steps in modules.

2. Shows the HTTP method we will use to send the URI to API.

3. Shows the URI we will call to the API.

4. Shows the 'Send' button which when clicked send the request to vSphere APIs.

5. Shows the JSON data structure for calls that require input

6. Shows the response from the API once the Send button is clicked
7. Shows / Hides the Action Sidebar, Note: You might need to click on this tab if the REST actions tab is not visible due to mismatch in screen resolution between HOL console and your PC.

Execute a Sample Workflow - Login to vCenter Server.

The following POST request will connect to the vCenter server. This action will act as the prerequisite for all the following actions

1. Select the “Login to vCenter” action

2. Click the “Send” button to send the API request, the user name and password are already added to this request as a header.

3. You will see that we get a token returned successfully from the API call.

Note: Credentials to connect to the vCenter Server have been pre imported into the POSTMAN client
4. Click on the "Eye" icon to see the environment variables with their values.

Execute a Sample Workflow - Show Datacenter Summary.

The following GET request will report the datacenter summary.

1. Select the **"Show Datacenter Summary"** action
2. Click the “Send” button to send the API request.

3. You will see the ID of our Datacenter and the friendly name.

4. Make a note of the ID, we will use this in our next step.

If you look at the method it is performing a GET, the URI for this call starts with **https://{vc}/rest** which is the location of the API, after this it uses /vccenter as this is a vcenter object and then /datacenter.

This basic format is used throughout the API and as you use the API more you should be able to guess some of the conventions without always referring to the API documentation.

**Execute a Sample Workflow - Show Datacenter Details.**

The following **GET** request will report the datacenter details for the **datacenter id** we made a note of in the previous step.

1. Select the “**Show Datacenter Details**” action.
Take a note of the URI, this time it includes the same URI but appended is the **datacenter ID**, now we can be more specific about the details we want to get from the API, you will notice this same convention as we follow the other actions in this list for items such as datastores, folders, VMs etc.

2. Validate the **datacenter ID** in the URI (2) matches the picture above: "datacenter-21".

3. Click the **Send** button to send the API request.

4. You will notice that the detailed information of the Datacenter is returned, in this case you will see the default hidden folders which are created when a datacenter is created which can be used for placement later.

**Execute a Sample Workflow - Show Datastore Summary.**

The following **GET** request will retrieve the datastores managed by the vCenter server.

1. Select the **"Show Datastore Summary"** action
2. Click the **"Send"** button to send the API request.
3. You will see that we are returned with a detailed information of the datastores.
Execute a Sample Workflow - Show all Folders.

The following **GET** request will list the details on the Folders.

1. Select the **“Show the Folders”** action

2. Click the **“Send”** button to send the API request.

3. You will notice that the detailed information of all the folders on the vcenter server are returned.
Execute a Sample Workflow - Find a folder for new VM using filter

The following **GET** request will retrieve the default folder for VM which gets created in the vCenter.

1. **Select the “Find a folder for new VM using filter” action**

   You will notice the URI has changed to include a filter which filters firstly on the type of folder being “VIRTUAL_MACHINE” and secondly on the name of the folder being “Discovered Virtual Machine” this is the default folder which gets created by vCenter.

2. **Click the “Send” button to send the API request.**

3. **You will see that we get returned with an folder ID, we will make use of this ID in the next steps.**
Execute a Sample Workflow - Find a datastore for new VM using filter

The following **GET** request will retrieve the default folder for datastore which gets created on the vCenter server.

1. Select the **“Find a datastore for new VM using filter”** action
2. Click the **“Send”** button to send the API request.
3. You will see that we are returned with a datastore ID, we will make use of this ID in the next steps.
Execute a Sample Workflow - Find a Resource Pool for new VM using filter

The following GET request will retrieve the default resourcepool ID for resourcepool which gets created on the vCenter server.

1. Select the “Find a Resource Pool for new VM using filter” action
2. Click the “Send” button to send the API request.
3. You will see that we are returned with a resourcepool ID, we will make use of this resourcepool ID in the next steps.
Execute a Sample Workflow - Create a simple VM

The following **POST** request will create a simple VM. This action will need inputs for the datastore ID, folder ID and the resourcepool ID we reported from the previous step.

1. **Select the “Create a simple VM” action**

Before you run this workflow, Click on the "**Body**" tab you will see that the how the method is now POST and that we have a body which includes some simple details as to where to place our VM and what guest OS to use. See also that we are not specifying a name here, this really has been altered to take the bare minimum required to create a VM for this API, of course we can specify more but at the least we currently need to specify these items. Update the details for the **datastore ID**, **folder ID** and the **resourcepool ID** we reported from the previous steps

2. **Click the “Send” button to send the API request.**

3. **The send action should complete quickly. Click on the Authorization tab again, and you will see a lower section that we get a ID returned for the virtual machine which gets created. (Which may be different that vm-242)**
Execute a Sample Workflow - Create a Fully Configured VM

The following POST request will create a fully configured VM. This action will need inputs for the datastore ID, folder ID and the resourcepool ID we reported from the previous step.

Return to the POSTMAN tool

1. Select the “Create fully configured VM” action

Before you run this workflow, Click on the "Body" tab, you will see that the how the method is now POST click on body and you will see that we have a body which includes an example of addition items which can be used to create a more fully featured VM. See also that we are not specifying a name here, this really has been altered to take the bare minimum required to create a VM for this API, of course we can specify more but at the least we currently need to specify these items. review the details for the datastore ID, folder ID and the resourcepool ID we reported from the previous steps

2. Click the “Send” button to send the API request.
3. The send action should complete quickly. Click on the Authorization tab again, you will see that we get an ID returned for the virtual machine which gets created.

**Execute a Sample Workflow - Show all VMs Summary.**

![Postman Interface](image)

The following **GET** request will retrieve details of VMs managed by the vCenter server.

1. Select the **“Show all VMs Summary”** action
2. Click the **“Send”** button to send the API request.
3. You will see that the detailed information of the VMs managed by the vCenter server are returned.
4. Click the X to close postman
Verify the VM creation with vCenter WebClient

Once we have the simple VM and the fully configured VM created via the Postman Client, now let us login to the vCenter server vcsa-01b and verify that they are present.

1. Launch Google Chrome by clicking the "Chrome" icon on the taskbar.

Verify the VM creation with vCenter H5 Client - (Continued)

1. Click on "RegionB vCenter" in the bookmarks tab and log in with the credentials of username as "administrator@corp.local" and password of "VMware1!"
Verify the VM creation with vCenter H5 Client - (Continued)

1. Click on "VMs and Templates" tab
2. Select the vcsa-01b.corp.local vCenter
3. Click on "VMs" tab
4. Verify the existence of the virtual machines
5. Minimize the Chrome Browser window.
Exploring DCLI features and workflow

Datacenter Command Line Interface (DCLI) is a command line designed to provide automation and a troubleshooting for features exposed via the vSphere REST API. This is just one of the interfaces which can be used to work with this API, others include PowerCLI, vRealize Orchestrator and various SDKs.

During the following steps you will learn how the knowledge of the API will easily transfer to the CLI and vice-versa, you will see that this CLI can be used as a multi platform CLI to work with vSphere and achieve basic automation and troubleshooting workflows.

DCLI comes by default with the VCSA, vCenter for Windows and is available as part of vCLI.

Connect to the vCenter server using DCLI and Windows Command Prompt.

1. Click on the Command Prompt icon on the taskbar

Connect to the vCenter server using DCLI and Windows Command Prompt Continued.

1. Type in the following command to open the DCLI directory

```
C:\Users\Administrator>cd C:\LabFiles\vmware-dcli
C:\LabFiles\vmware-dcli>dcli.exe +server vcsa-01b +skip-server-verification
Welcome to VMware Datacenter CLI(DCLI)
usage: dcli <namespaces> <command>
To enter interactive shell mode: dcli +interactive
To specify an alternate server: dcli +server <server>
For detailed help please use: dcli --help
```

Root namespace
Available Namespaces:
com com namespace
C:\LabFiles\vmware-dcli>

1. Type in the following command to open the DCLI directory
2. Type in the following command to use DCLI to access the VMware Datacenter CLI (DCLI)

dcli.exe +server vcsa-01b +skip-server-verification

3. Hit Enter, you might be asked a user name and password, type in administrator@corp.local for the username and VMware1! for the password. Say yes if you are asked to save the credentials. Once connected and at the appliance shell, you will also see a list of namespaces that are available to use in DCLI

Note: DCLI can be used in both interactive and single command mode

Explore DCLI help

C:\LabFiles\vmware-dcli\dcli.exe --help

usage: dcli ( +server SERVER | +interactive | +prompt PROMPT | +skip-server-verification | +cacert-file CERT_FILE ) [ +more ] [ +loglevel { debug, info, warning, error } ] [ +username USERNAME ] [ +credstore-file CREDSTORE_FILE ] [ +credstore-add | +credstore-remove | +credstore-list ] [ +session-manager SESSION_MANAGER ] [ args [ args ... ] ]

positional arguments:
  args          CLI command

optional arguments:
  +server SERVER  Specify the server IP address/DNS name (default: http://localhost/api)
  +interactive   Open a CLI shell to invoke commands
  +prompt PROMPT Prompt for cli shell (default: dcli> )
  +skip-server-verification  Skip server SSL verification process (default: False)
  +cacert-file CERT_FILE  Specify the certificate authority certificates for validating SSL connections (format: PEM) (default: )
  +more          Flag for page-wise output
  +formatter { simple, table, xml, json, html, csv }  Specify the formatter to use to format the command output
  +loglevel { debug, info, warning, error }  Specify the log verbosity (default: info)
  +username USERNAME  Specify the username for login (default: )
  +credstore-file CREDSTORE_FILE  Specify the dcli credential store file (default: C:\Users\Administrator\.dcli\dcli_credstore)
  +credstore-add  Store the login credentials in credential store
  +credstore-remove  Remove the login credentials from credential store
  +credstore-list  List the login credentials stored in credential store
  +session-manager SESSION_MANAGER  Specify the session manager for credential store remove operation
1. Type in the following command to explore DCLI help and hit enter key.

   dcli.exe --help

2. You will see that the options can be used by providing a + to the DCLI command, these include the ability to output in many formats, run interactively, provide a server to connect to and many more options, familiarize yourself with some of the functionality.

List VMs using DCLI

1. Type in the following command to list VMs and hit enter key. This command directly maps to GET /rest/vcenter/vm. This will return a list of vms which are being managed by the vcenter server.

   This command may ask you to provide credentials, remember this is being done through vCenter so the administrator@corp.local username and password of VMware1!

   dcli.exe +server vcsa-01b +skip-server-verification com vmware vcenter vm list

2. You will see a list of VMs managed by the vCenter server.
Explore DCLI interactive mode and Report VM Data in different formats
```
Administrator: Command Prompt - dcli.exe -server vcssa-01b -skip-server-verification +i
c:\LabFiles\vmware-dcli\dcli.exe +server vcssa-01b +skip-server-verification +i
Welcome to VMware Datacenter CLI(<DCLI>)

usage: <namespaces> <command>
To auto-complete and browse DCLI namespaces: (TAB)
If you need more help for a command: vcenter vm --help
If you need more help for a namespace: vcenter vm --help
For detailed information on DCLI usage visit: http://vmware.com/go/dcli

dcli> vm list
<table>
<thead>
<tr>
<th>Name</th>
<th>PowerState</th>
<th>CpuCount</th>
<th>MemorySizeMiB</th>
</tr>
</thead>
<tbody>
<tr>
<td>vm-346 kms-01b.corp.local</td>
<td>Powered_On</td>
<td>12</td>
<td>1024</td>
</tr>
<tr>
<td>vm-347 kms-02b.corp.local</td>
<td>Powered_On</td>
<td>12</td>
<td>1024</td>
</tr>
<tr>
<td>vm-382 esx-01c.corp.local</td>
<td>Powered_Off</td>
<td>12</td>
<td>14096</td>
</tr>
<tr>
<td>vm-401 w10-base-01b.corp.local</td>
<td>Powered_Off</td>
<td>11</td>
<td>14096</td>
</tr>
</tbody>
</table>

```

```
deli> vm list +formatter simple

vm-346
Name: kms-01b.corp.local
PowerState: Powered_On
CpuCount: 2
MemorySizeMiB: 1024

vm-347
Name: kms-02b.corp.local
PowerState: Powered_On
CpuCount: 2
MemorySizeMiB: 1024

vm-382
Name: esx-01c.corp.local
PowerState: Powered_Off
CpuCount: 2
MemorySizeMiB: 4096

vm-401
Name: w10-base-01b.corp.local
PowerState: Powered_Off
CpuCount: 1
MemorySizeMiB: 4096

```

```
deli> vm list +formatter simple

vm-423
Name: Full Configured VM
PowerState: Powered_Off
CpuCount: 1
MemorySizeMiB: 16

```
Next you will use Interactive mode, this gives us added features such as shortening of command namespace names, tab completion of names, tab completion of required options and more.

1. Type in the following command to use DCLI in interactive mode, you will notice the prompt change to dcli>, hit

   `dcli.exe +server vcsa-01b +skip-server-verification +i`

   You might be asked a user name and password, type in administrator@corp.local for the username and **VMware1!** for the password. Say yes if you are asked to save the credentials. Once connected and at the appliance shell, you will also see a list of namespaces that are available to use in DCLI. To show we are now in interactive mode and we will also get a reminder of the namespaces we can use.

2. Type in the following command to list the vms managed by the vCenter in the DCLI interactive mode, hit the enter key.

   `vm list`

   You can also use the options inside of interactive mode

3. Type in the following command to list the vms managed by the vCenter in simple format, hit the enter key.

   `vm list +formatter simple`

4. Type in the following command to list the vms managed by the vCenter in json format, hit the enter key.

   `vm list +formatter json`
Explore available namespaces and help

DCLI also allows you to create and delete VMs,

1. Type in the following command to get a list of available commands, hit the enter key. you will see the available commands listed are **hardware**, **power**, **list**, **delete**, **create**, and **get**
2. To get more help on the command append a `--help` at the command end and observe the syntax result obtained. Type in the following command:

```
vm list --help
```

## List Folders, Datastores and ResourcePools with DCLI

Type in the following commands to report a list of **folders**, **datastores** and **resourcepools** available in the cluster using **DCLI**.

1. Type in the following command to report a list of folder

```
folder list
```

2. Type in the following command to report a list of datastores

```
datastore list
```

3. Type in the following command to report a list of resourcepools

```
resourcepool list
```
Export VM Data reported by DCLI in CSV format and export it.

DCLI allows you to export the VM data in csv format and also export it to a CSV file. You need to exit from the DCLI interactive mode by typing in `exit`.

1. Type in the following command to Report VM data in CSV format and export it to a CSV file named `vmlist.csv`:

   ```
   dcli.exe +server vcsa-01b +skip-server-verification com vmware vcenter vm list +formatter csv > vmlist.csv
   ```

2. Open up the file in notepad and view its content with the following command:

   ```
   notepad vmlist.csv
   ```
Create VMs using DCLI Shell script

We can bundle the commands we learned in the previous set of steps to create a shell script which will go ahead and create a set of VMs using DCLI commands.

1. Click on the "Putty" icon placed on the "Windows Taskbar".
2. The Putty client should open up.

Create VMs using DCLI Shell script - (Continued)

We can bundle the commands we learned in the previous set of steps to create a shell script which will go ahead and create a set of VMs using DCLI commands.

1. Login to vcsa-01b using the putty client, type in vcsa-01b in the host name tab.
2. Click on **Open**.

Create VMs using DCLI Shell script - (Continued)

![Image]

1. Next login as **root**, and hit enter, you will not be prompted for a password

Create VMs using DCLI Shell script - (Continued)

![Image]

Next lets make sure and clean up any existing credentials in the DCLI credential store.

1. Execute the below command to clear up the cred store, and hit enter key

   ```
   dcli +server vcса-01b +skip-server-verification com vmware vcenter vm list +credstore-remove
   ```

**Note:** You might see an error which states "Error: Couldn't find credstore entry. Please pass correct user and server values", this error message can be ignored.
Create VMs using DCLI Shell script - (Continued)

Next we need to add the credentials to connect to the vcenter server in the DCLI credential store.

1. Type in the following command and type in the username as `administrator@corp.local` and password as `VMware1!`, and hit enter

   ```
dcli +server vcса-01b +skip-server-verification com vmware vcenter vm list +credstore-add
   ```

   The credentials provided are now saved for the session

Create VMs using DCLI Shell script - (Continued)

We can bundle the commands we learned in the previous set of steps to create a shell script which will go ahead and create a set of VMs using DCLI commands.

```
1. Type in the following command to change your present working directory to root

```bash
cd ~
```

Next explore the contents of the shell script

2. Type in the following command to see the contents of `createvms.sh` script

```bash
cat createvms.sh
```

The contents of the script reported on the screen contains the DCLI script written to create 10 test VMs and list them.

--- VM List ---

<table>
<thead>
<tr>
<th>VM</th>
<th>Name</th>
<th>PowerState</th>
<th>CpuCount</th>
<th>MemorySizeMB</th>
</tr>
</thead>
<tbody>
<tr>
<td>vm-116</td>
<td>Full Configured VM</td>
<td>POWERED_OFF</td>
<td>1</td>
<td>1024</td>
</tr>
<tr>
<td>vm-117</td>
<td>LoopVM0</td>
<td>POWERED_OFF</td>
<td>1</td>
<td>1024</td>
</tr>
<tr>
<td>vm-118</td>
<td>LoopVM1</td>
<td>POWERED_OFF</td>
<td>1</td>
<td>1024</td>
</tr>
<tr>
<td>vm-119</td>
<td>LoopVM2</td>
<td>POWERED_OFF</td>
<td>1</td>
<td>1024</td>
</tr>
<tr>
<td>vm-120</td>
<td>LoopVM3</td>
<td>POWERED_OFF</td>
<td>1</td>
<td>1024</td>
</tr>
<tr>
<td>vm-121</td>
<td>LoopVM4</td>
<td>POWERED_OFF</td>
<td>1</td>
<td>1024</td>
</tr>
<tr>
<td>vm-122</td>
<td>LoopVM5</td>
<td>POWERED_OFF</td>
<td>1</td>
<td>1024</td>
</tr>
<tr>
<td>vm-123</td>
<td>LoopVM6</td>
<td>POWERED_OFF</td>
<td>1</td>
<td>1024</td>
</tr>
<tr>
<td>vm-124</td>
<td>LoopVM7</td>
<td>POWERED_OFF</td>
<td>1</td>
<td>1024</td>
</tr>
<tr>
<td>vm-125</td>
<td>LoopVM8</td>
<td>POWERED_OFF</td>
<td>1</td>
<td>1024</td>
</tr>
</tbody>
</table>

Create VMs using DCLI Shell script - (Continued)
Now, let's invoke the script and create 10 VMs.

1. Type in the following command to invoke the DCLI shell script

   ./createvms.sh

Observe 10 VMs created using the script, once the script finishes execution you would be able to see the VMs created and listed in the putty session.

**Verify the VM creation with vCenter H5 Client**

Now, let us login to the vCenter server vcsa-01b and verify that the 10 VMs have been created.

1. Launch Google Chrome by clicking the "Chrome" icon on the taskbar.

**Verify the VM creation with vCenter H5 Client - (Continued)**

1. Click on "RegionB vCenter" in the bookmarks tab and log in with the credentials of username as "administrator@corp.local" and password of VMware1!
Verify the VM creation with vCenter H5 Client - (Continued)

1. Click on "VMs and Templates" tab
2. Select the vcsa-01b.corp.local vCenter
3. Click on "VMs" tab
4. Verify the existence of the virtual machines
5. Minimize the Chrome Browser window.
Module 2 - Python and the vSphere API (45 minutes)
Introduction

In this module, you will learn how to work with Python and vSphere API.

If you have not already reviewed it, the Lab Introduction contains details about Rainpole Systems (our example company), as well as important information on text entry, multiple language support and on-screen keyboard configuration. If desired, you can click here to review this information again.

This Module contains the following lessons:

- **Exploring the Python SDK Directory Structure**: In this lesson you will explore and learn about the contents of the Python SDK directory.
- **Managing Virtual Machine using Python and vSphere Automation API**: Learn how to automate Virtual Machine management using Python and vSphere Automation API.

Introducing VMware vCenter REST API

vCenter REST API is a new developer friendly, simplified API which allows for Virtual Machine creation, modification and deletion via a consistent set of developer and automation tooling. This will allow for consistent interaction with the API via the tools which make sense to the developer community or someone automating VMware products.

The focus of this lab session will be on vCenter REST API Calls.

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
Exploring the Python SDK Directory Structure

The VMware vCloud Suite SDK for Python enables programmatic access to vSphere. Access to capabilities exposed through the vSphere Web Services API is done using the open-source pyVmomi project. The SDK includes python libraries for accessing new features like Content Library and existing features like Tagging which did not support a public API prior to latest release of vSphere. The SDK contains samples for the features mentioned above as well as samples to demonstrate how to interoperate with vSphere APIs. If you want to look at the vSphere API samples,

The VMware vCloud Suite SDK for Python is intended for the following audiences:

- Developers who want to write new application(s) for future releases of vSphere.
- Developers migrating an existing vSphere application to use new features.

The vCloud Suite Client SDK for Python provides client bindings, documentation, samples, and WSDL files. The following table shows the client directories and their contents.

In this module we will explore the content of the Python SDK directory structure.

Exploring the Python SDK directory structure

1) Click the Windows Explorer icon on the taskbar

Exploring the Directory Structure for Python SDK - bin Directory
The bin directory consists of the following script files

- **Main.Py** - This is the python script that runs through all the setup and samples and is used to pre-configure the environment which would act as our testbed.

*Note: The above script is preconfigured in the HOL setup.*

- **run_sample.bat** - To execute the Python command based sample scripts on windows environment, you will need to use the run_sample.bat file.
- **run_sample.sh** - To execute the Python command based sample scripts on a UNIX/mac OS X environment, you will need to use the run_sample.sh file.

1. Explore the contents of the directory 'C:\LabFiles\VMware-vCloud-Suite-SDK-Python-6.5.0\client\bin' in the "Windows Explorer"

### Exploring the Directory Structure for Python SDK - lib Directory

For python developers, client libraries are supplied for testing and development purposes. All the supplied libraries are located under lib directory.

The lib directory consists of the following python libraries

- **vapi_client_bindings-2.5.0.zip** - vAPI runtime responsible for serialization/deserialization of objects and wire protocol.
- **vapi_common_client-2.5.0.zip** - This library consists of the vAPI common client code.
- **vapi_runtime-2.5.0.zip** - This library consists of client stubs for vCloud Suite APIs.

1. Explore the contents of the directory 'C:\LabFiles\VMware-vCloud-Suite-SDK-Python-6.5.0\client\lib' in the "Windows Explorer"
Exploring the Directory Structure for Python SDK - samples Directory

The vCloud Suite Python SDK samples are located in the client sample directory
'C:\LabFiles\VMware-vCloud-Suite-SDK-Python-6.5.0\client\samples\src\com\vmware\vcloud\suite\sample'.

The vCloud Suite Python SDK samples use a framework to facilitate different aspects of using the samples. Some of the framework capabilities are:

- Command line argument parsing.
- Specifying mandatory and optional arguments for a sample.
- Sample setup, execution and post-execution cleanup.
- Information about samples.

The contents of the sub directory are as explained below

- The directory common consists of the samples common classes and abstractions.
- The directory 'VIM\Helpers' consists of samples and utilities for accessing and manipulating VC objects using pyVmomi.
- The directory lookupservice contains samples for service discovery sample using lookup service APIs
- The directory workflow consists of various vAPI work flow samples
- The directory inventory consists of samples for inventory APIs for retrieving information about vCenter datastore and network objects.

1. Navigate to the directory listed above and explore the contents
Explore the contents of wsd1 Directory

The vCloud Suite SDK for Python samples use the vCloud Suite Lookup Service to obtain the URLs for other vCloud Suite services (SSO, vAPI, VIM, SPBM, etc.). The SDK contains the Lookup Service WSDL files. The samples use the python SUDS client for accessing the lookup service.

The Lookup Service WSDL files are located in the following SDK directory 'C:\LabFiles\VMware-vCloud-Suite-SDK-Python-6.5.0\client\wsdl'

The wsd1 directory consists of the following files

- **lookup.wsdl** - Lookup Service WSDL files.
- **lookupservice.wsdl** - Lookup Service WSDL files.

1. Explore the contents of the directory 'C:\LabFiles\VMware-vCloud-Suite-SDK-Python-6.5.0\client\wsdl' in the "Windows Explorer"
Managing Virtual Machine using Python and vSphere Automation API

In this module we will look at some sample python scripts which will illustrate the usage of Python with vSphere API for automating some common administrative actions.

We will also use PowerCLI to compare the results obtained via the script execution.

In this module we will explore the following.

- **testbed.py** - This script file to build the Test bed consists of the pre-requisites configuration which need to be present

- **create_vms.py** - This python script file demonstrate how to create create a default, basic and complex VM using vsphere automation API.

- **memory.py** - This python script file demonstrate how to change the memory setting for a VM using vsphere automation API.

- **cpu.py** - This python script file demonstrate how to change the CPU setting for a VM using vsphere automation API.

- **delete_vms.py** - This python script file demonstrate how to delete the test vms created using vsphere automation API.
Explore the script file TestBed.Py

```python
config = {}
config["SERVER"] = "ESX-01.corp.local"
config["USERNAME"] = "administrator@corp.local"
config["PASSWORD"] = "VMware1!

config["ESX_HOST1"] = "esx-01b.corp.local"
config["ESX_HOST2"] = "esx-01b.corp.local"
config["ESX_USER"] = "root"
config["ESX_PASS"] = "VMware1!

config["USE_NFS"] = False
config["NFS_HOST"] = ""
config["NFS_REMOTE_PATH"] = "/store1"
config["NFS_DATASTORE_NAME"] = "RegionB01-ISCSI01-COMP01"

config["ESX_HOST1_VNFS_DATASTORE"] = "RegionB01-ISCSI01-COMP01"
config["ESX_HOST2_VNFS_DATASTORE"] = "RegionB01-ISCSI01-COMP01"

config["DATACENTER1_NAME"] = "RegionB01"
config["DATACENTER2_NAME"] = "RegionB01"

config["VM_FOLDER1_NAME"] = "Discovered virtual machine"
config["VM_FOLDER2_NAME"] = "Discovered virtual machine"

config["CLUSTER1_NAME"] = "RegionB01-COMP01"

config["VDI_SWITCH1_NAME"] = "ESXi-RegionB01-vDS-COMP"
config["VDFORTGROUP1_NAME"] = "ESXi-RegionB01-vDS-COMP"

config["STDGROUP_NAME"] = "VM-RegionB01-vSS-COMP"

# The main datacenter and datastore that will be used for the VM tests
config["VM_DATACENTER_NAME"] = config["DATACENTER2_NAME"]
config["VM_DATASTORE_NAME"] = config["NFS_DATASTORE_NAME"]
```

This script file consists of the python script which needs to be execute to create the test bed, also note that this script has been pre-executed for this lab to build the test-bed, we will go through the contents of the script.

1. The following image consists of a extract of the contents of this file, the following mandatory settings need to be present in the file, which will be used to build the test-bed

```python
config["SERVER"] = ""
config["USERNAME"] = ""
config["PASSWORD"] = ""
config["ESX_HOST1"] = ""
config["ESX_HOST2"] = ""
config["ESX_USER"] = ""
```
Create VMs in the test-bed by using `create_vms.py`

1. Click on the Command Prompt icon on the taskbar
Create VMs in the test-bed by using create_vms.py

Continued

1. Navigate to the python bin directory 'C:\LabFiles\VMware-vCloud-Suite-SDK-Python-6.5.0\client\bin'
2. Type in the following command to create a set of VMs and hit enter.

```
run_sample.bat ../samples/src/com/vmware/vcloud/suite/sample/vcenter/vm/create_vms.py
```

3. The above command creates a set of 3 sample VMs, named **default**, **basic** and **complex** VM using vsphere automation API.

**Note:** The command should execute successfully and create three VMs.

### Verify the VM creation with vCenter WebClient

Once we have the VMs created via the python script now, let us login to the vCenter server vcsa-01b and verify that they are present.

1. Launch Google Chrome by clicking the "Chrome" icon on the taskbar.

### Verify the VM creation with vCenter H5 Client - (Continued)

1. Click on "RegionB vCenter" in the bookmarks tab and log in with the credentials of username as "**administrator@corp.local**" and password of **VMware1!**
Verify the VM creation with vCenter H5 Client -
(Continued)

1. Click on "VMs and Templates" tab
2. Select the vcsa-01b.corp.local vCenter
3. Click on "VMs" tab
4. Verify the existence of the virtual machines, you may need to scroll down in VM window.
5. Minimize the Chrome Browser window.
Modify the CPU properties for VMs in the test-bed by using cpu.py

The Python API's also allow you to modify certain CPU properties.

Open up the command prompt.

1. Type in the following command to alter the CPU properties for the Virtual Machines
2. Once the command completes its execution it would effectively enable the CPU hot add, then disable the CPU hot remove and also modify the number of cores per socket in the VMs using vsphere automation API.

Modify the VM Memory setting in the test-bed by using memory.py

```
Using uc host specified in testbed.py
Using uc user specified in testbed.py
Using uc password specified in testbed.py
Using vm name <VM_NAME_DEFAULT> specified in testbed.py

# Get current Memory configuration
vm.hardware.Memory.get(vm-185) -> Info(
    hot_add_enabled=False,
    hot_add_increment_size_mib=None,
    hot_add_limit_mib=None,
    size_mib=4096,
)

# Update memory size_mib field of Memory configuration
vm.hardware.Memory.update(vm-185, { 'hot_add_enabled': False, 'size_mib': 8192 })
vm.hardware.Memory.get(vm-185) -> Info(
    hot_add_enabled=False,
    hot_add_increment_size_mib=None,
    hot_add_limit_mib=None,
    size_mib=8192,
)

# Update hot_add_enabled field of Memory configuration
vm.hardware.Memory.update(vm-185, { 'hot_add_enabled': True, 'size_mib': None })
vm.hardware.Memory.get(vm-185) -> Info(
    hot_add_enabled=True,
    hot_add_increment_size_mib=None,
    hot_add_limit_mib=None,
    size_mib=8192,
)

# Revert Memory configuration
vm.hardware.Memory.update(vm-185, { 'hot_add_enabled': False, 'size_mib': None })
vm.hardware.Memory.get(vm-185) -> Info(
    hot_add_enabled=False,
    hot_add_increment_size_mib=None,
    hot_add_limit_mib=None,
    size_mib=4096,
)
```
Open up the command prompt.

1. Type in the following command to alter the Memory for the Virtual Machines

   run_sample.bat ../samples/src/com/vmware/vcloud/suite/sample/vcenter/vm/hardware/memory.py

2. Once this command executes this would effectively enable the Memory hotadd and also modify the VM memory to 8GB and revert the memory of the VM back to from 4GB using the automation API.

Delete VMs in the test-bed by using delete_vms.py

Open up the command prompt.

1. Type in the following command to alter the Memory for the Virtual Machines

   run_sample.bat ../samples/src/com/vmware/vcloud/suite/sample/vcenter/vm/delete_vms.py

2. Once the above command executes successfully, it will delete the sample VMs we created earlier using the automation API.
Delete VMs in the test-bed by using delete vms.py - (Continued)

1. Maximize the Chrome Browser window (Not Shown)
2. Under Recent Tasks, three Delete virtual machine tasks will be listed.
3. Type sample in the search window and it will return no results.
Module 3 - Java and the vSphere API (45 minutes)
Introduction

In this module, you will learn how to work with Java and vSphere API.

If you have not already reviewed it, the Lab Introduction contains details about Rainpole Systems (our example company), as well as important information on text entry, multiple language support and on-screen keyboard configuration. If desired, you can click here to review this information again.

This Module contains the following lessons:

- **Exploring the Java SDK directory structure**: In this lesson you will explore and learn about the contents of the Java SDK directory.
- **Managing Virtual Machine using Java and vSphere Automation API**: Learn how to automate Virtual Machine management using Java and vSphere Automation API.

Introducing VMware vCenter REST API

vCenter REST API is a new developer friendly, simplified API which allows for Virtual Machine creation, modification and deletion via a consistent set of developer and automation tooling. This will allow for consistent interaction with the API via the tools which make sense to the developer community or someone automating VMware products.

The focus of this lab session will be on vCenter REST API Calls.

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 Consumarization 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
Exploring the Java SDK directory structure

The VMware vCloud Suite SDK for Java enables programmatic access to vSphere. Access to capabilities exposed through the vSphere Web Services API is done using the vSphere Management SDK. The SDK includes java libraries for accessing new features like Content Library and existing features like Tagging.

The SDK contains samples for the features mentioned above as well as samples to demonstrate how to interoperate with vSphere APIs. If you want to look at the vSphere API samples, please refer to the vSphere Management SDK.

Intended Audience

The VMware vCloud Suite SDK for Java is intended for the following audiences:

- Developers who want to write new application(s) for vSphere.
- Developers migrating an existing vSphere application to use new features.

The client SDK contains Java libraries for the new features and the vSphere API, API Reference documentation and sample code. See the Client SDK README.

Exploring the Java SDK directory structure - Lib

1) Click the Windows Explorer icon on the taskbar
Exploring the Java SDK directory structure - Lib

The Lib directory consist of the Client SDK, samples, other jar files needed by the samples

1. Browse to the directory 'C:\LabFiles\VMware-vSphere-Automation-SDK-Java\client\lib' and explore the contents

Exploring the Java SDK directory structure - Samples

The samples directory consists of vCloud Suite Client SDK Java client samples for various features and workflows.
1. Browse to the directory 'C:\LabFiles\VMware-vSphere-Automation-SDK-Java\client\samples\java\vmware\samples\vcenter\vm' and explore the contents.

You will see a couple of files listed, we will use **CreateVM.java** for our sample tests.

**Close Windows Explorer**

1. Click the **X** in the top right corner of Windows Explorer to close the window.
Managing Virtual Machine using Java and vSphere Automation API

In this module we will look at a sample java script which will illustrate the usage of Java with vSphere API for automating vm creation and deletion.

You can run the samples from the command line. The SDK provides the command line scripts run.sh (Linux) and run.bat (Windows).

As our Main Console VM is windows, we will use the run.bat command to execute the script.

In this module we will explore the following.

CreateVM.Java - This script file will create and subsequently delete VMs from the vcenter server.

Creating VMs using CreateVM.java

1. Click on the Command Prompt icon on the taskbar

Creating VMs using CreateVM.java

1. Navigate to the directory 'C:\LabFiles\VMware-vSphere-Automation-SDK-Java\client\bin'
2. Execute the following command
run.bat vmware.samples.vcenter.vm.CreateVM --server vcsa-01b.corp.local --username administrator@corp.local --password VMware1! --datacenter RegionB01 --cluster RegionB01-COMP01 --datastore RegionB01-ISCSTI01-COMP01 --distributedportgroup VM-RegionB01-vDS-COMP --standardportgroup "VM-RegionB01-vSS-COMP" --isodataloastorepath "[RegionB01-ISCSTI01-COMP01] iso/Core-7.1.iso" --vmfolder "Discovered virtual machine" --cleardata false --resourcepool Resources

 Executing this command will create a set of sample VMs and on the esxi hosts using Java in conjunction with vsphere automation API.

 The command should execute successfully and create three VMs

 **Verify the VM creation with vCenter H5 Client**

 ![Chrome icon](image)

 Once we have the VMs created via the Java script, now let us login to the vCenter server vcsa-01b and verify that they are present.

 1. Launch Google Chrome by clicking the "Chrome" icon on the taskbar.

 **Verify the VM creation with vCenter H5 Client - (Continued)**

 ![RegionB H5 icon](image)

 1. Click on "RegionB vCenter" in the bookmarks tab and log in with the credentials of username as "administrator@corp.local" and password of VMware1!
Verify the VM creation with vCenter H5 Client - (Continued)

1. Click on "VMs and Templates" tab
2. Select the `vcsa-01b.corp.local` vCenter
3. Click on "VMs" tab
4. Verify the existence of the virtual machines
5. Minimize the Chrome Browser window.
Module 4 - API Explorer (30 minutes)
Introduction

In this module, you will learn about the API Explorer.

The API Explorer allows users to connect to the API endpoint and then browse and interact with available REST based APIs and their calls. This includes available parameters, expected responses, what response status codes may mean, and much more. If you've happened to use the Managed Object Browser, or MOB, think of that but on steroids!

This Module contains the following lessons:

- **Accessing the API Explorer**: In this lesson you will show you how to get to the API Explorer.
- **Browsing the API Explorer**: Learn how to use the API Explorer.
Accessing the API Explorer

In this module we will look at the API Explorer

Accessing the API Explorer is incredibly easy. It's available on any API endpoint whether that be a vCenter server (appliance or Windows based) or external PSC appliance.

Open API Explorer in Chrome

Launch Chrome

1. Launch Google Chrome by clicking the "Chrome" icon on the taskbar.

Click on Bookmark for API Explorer

1. Click on "API Explorer" in the bookmarks tab.
The API Explorer Interface

After having loaded the API Explorers interface, its time to start looking around and seeing whats available. There are two items that should jump out immediately, the login option and the Select API dropdown.

- Appliance  handles calls to the underlying VMware appliance
- CIS Common Infrastructure Services, handles calls involving tagging
- Content  handles calls involves the Content Library
- vAPI vSphere API, handles calls regarding the API endpoint
- vCenter  handles calls to involving the vCenter

1. Select vCenter from API Drop Down.
2. Click on **Cluster**

1. Click on **GET /vcenter/cluster** operation to expand and reveal example code.

We can see the path to make the call and the method. However, the API Explorer also gives some other information such as the expected response class, parameters and potential response messages. Clicking on those items expands and collapses additional information. (Response Class header) Scroll down to reveal more information.
My favorite part is right below those areas, the **Try It Out!** button. Clicking on that button performs the operation directly within the browser. It then provides a bunch of information such as an example cURL statement, the requests URL, as well as the requests body, status code, and headers.

1. Click on the **Try It Out!** button

The API Explorer is a great way to orient to and discover vSpheres REST based APIs. Not only does it provide lots of information and additional context around the API calls, it even helps users start making those calls.
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-1811-06-SDC

Version: 20170920-123528