Mac and Linux Users, Don't Despair: PowerCLI Is There!

Alan Renouf, VMware, Inc.
Luc Dekens, Eurocontrol
Disclaimer

This presentation may contain product features or functionality that are currently under development.

This overview of new technology represents no commitment from VMware to deliver these features in any generally available product.

Features are subject to change, and must not be included in contracts, purchase orders, or sales agreements of any kind.

Technical feasibility and market demand will affect final delivery.

Pricing and packaging for any new features/functionality/technology discussed or presented, have not been determined.
Introduction Video
PowerCLI on multiple OS versions
Introduction

Alan Renouf
Sr. Product Line Manager, VMware Cloud on AWS
Personal Blog: http://virtu-al.net/
Twitter: @alanrenouf

Luc Dekens
Systems Engineer
Personal Blog: https://lucd.info
Twitter: @LucD22
Agenda

- Introduction
- PowerShell Core
- PowerCLI Core
- Porting your scripts
- Side-by-Side
- New Possibilities
- Chef InSpec
- Q/A
A new playing field
PowerShell Everywhere

Main goal = Multi-platform

At a price
• Some cmdlets are not there
• .Net is different

Switching
• Price to pay
• Converting scripts
• Advantage
• Side-by-side (Windows only)
PowerShell Core
PowerShell Core
Aka PowerShell v6

Open Source
Based on .Net Core
Common IDE = Visual Studio Code editor
PowerShell Core
Aka PowerShell v6

Background Jobs

```
PS C:\> Get-ChildItem -Path c:\ -Recurse &
```

<table>
<thead>
<tr>
<th>Id</th>
<th>Name</th>
<th>PSJobTypeName</th>
<th>State</th>
<th>HasMoreData</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Job1</td>
<td>BackgroundJob</td>
<td>Running</td>
<td>True</td>
<td>localhost</td>
</tr>
</tbody>
</table>

VMworld 2018 Content: Not for publication or distribution
PowerShell Core
Aka PowerShell v6

Remote session over SSH

```
PS C:\> Enter-PSSession -HostName ubuntu1 -UserName luc -SSHTransport
luc@ubuntu1's password:
[ubuntu1]: PS /home/luc> $PSVersionTable

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSVersion</td>
<td>6.0.2</td>
</tr>
<tr>
<td>PSEdition</td>
<td>Core</td>
</tr>
<tr>
<td>GitCommitId</td>
<td>v6.0.2</td>
</tr>
<tr>
<td>OS</td>
<td>Linux 4.15.0-33-generic #36~16.04.1-Ubuntu SMP Wed Aug 15 17:21:05 UTC 2018</td>
</tr>
<tr>
<td>Platform</td>
<td>Unix</td>
</tr>
<tr>
<td>PSCompatibleVersions</td>
<td>{1.0, 2.0, 3.0, 4.0...}</td>
</tr>
<tr>
<td>PSRemotingProtocolVersion</td>
<td>2.3</td>
</tr>
<tr>
<td>SerializationVersion</td>
<td>1.1.0.1</td>
</tr>
<tr>
<td>WSManStackVersion</td>
<td>3.0</td>
</tr>
</tbody>
</table>

[ubuntu1]: PS /home/luc> exit
PS C:\>
```
PowerShell Core
Aka PowerShell v6

Where am I running?

Function Get-CorePlatform

Uses new $IsWindows / $IsLinux / $IsMacOS variables

Simplifies testing
function Get-CorePlatform {
    [cmdletbinding()]
    param()

    if ($IsWindows) {
        #
        #
    }
    elseif ($IsLinux -or $IsMacOS) {
        #
        #
    }
    else {
        #
        #
    }
}
VMware PowerCLI Multi-Platform

Everything is Get-Awesome!
PowerCLI

Where are we now?

PowerCLI 10.1
- Support for vSphere 6.7 (back to 5.5)
- Support for NSX-T 2.1
- New Module VMware.Vim
  - No cmdlets here?
  - This helps us release vSphere support updates FASTER!
- New Autodeploy cmdlets
  - Set-ScriptBundleAssociation
  - Remove-ScriptBundle

PowerCLI 10.1.1
- Updated Support for Horizon View 7.5

PowerCLI 10.2
- Support for NSX-T 2.2
PowerCLI Multi-Platform
VMware PowerCLI 10.*

Not Open Source (yet)
- Idea exists
- Same modules on PSGallery
  - Come with the 2 .Net flavours

Based on .Net Core
- Some Windows .Net calls missing

Ships with both .Nets for one module distribution

https://tinyurl.com/PowerCLIFreedom
## PowerCLI Differences

Which modules do I have?

<table>
<thead>
<tr>
<th>Module</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cis.Core</td>
<td>= Available in PowerCLI</td>
</tr>
<tr>
<td>Core</td>
<td>= Available in PowerCLI</td>
</tr>
<tr>
<td>Storage Utility</td>
<td>= Available in PowerCLI Multi-Platform</td>
</tr>
<tr>
<td>Vm</td>
<td>= Available in PowerCLI</td>
</tr>
<tr>
<td>Nsxt</td>
<td></td>
</tr>
<tr>
<td>Vds</td>
<td></td>
</tr>
<tr>
<td>Sdk</td>
<td></td>
</tr>
<tr>
<td>Vmc</td>
<td></td>
</tr>
<tr>
<td>Common</td>
<td></td>
</tr>
<tr>
<td>Deploy Automation</td>
<td>= Available in PowerCLI Multi-Platform</td>
</tr>
<tr>
<td>ImageBuilder</td>
<td></td>
</tr>
<tr>
<td>HA</td>
<td>= Deprecated in future release</td>
</tr>
<tr>
<td>License</td>
<td></td>
</tr>
<tr>
<td>HorizonView</td>
<td></td>
</tr>
<tr>
<td>PCloud</td>
<td>= Deprecated in future release</td>
</tr>
<tr>
<td>SrM</td>
<td></td>
</tr>
<tr>
<td>vROps</td>
<td></td>
</tr>
<tr>
<td>Vum Automation</td>
<td></td>
</tr>
<tr>
<td>Cloud</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** VMworld 2018 Content: Not for publication or distribution.
PowerCLI Core Differences

Cmdlets that no longer work

Datastore Provider
Get-VICredentialStoreItem
New-VICredentialStoreItem
Remove-VICredentialStoreItem
Get-VMHostHardware
Open-VMConsoleWindow
Porting your Scripts
Porting Credentials

New-VICredentialStoreItem

Based on Date Protection API (DPAPI)

Not in .Net Core

```powershell
PS /home/luc> $t = @{Name='abc'}
PS /home/luc> $t | Export-Clixml -Path ./test.xml
PS /home/luc> $t = @{Name='abc'; Pswd=Get-Credential}

PowerShell credential request
Enter your credentials.
User: luc
Password for user luc: ****
PS /home/luc> $t | Export-Clixml -Path ./test.xml
Export-Clixml : Unable to load DLL 'CRYPT32.dll': The specified
```
Porting
Credentials

Alternatives
• Use your own encryption
• Tool available on all platforms
• Tool that has an API
• For example: KeePassXC

Proxy functions for all VICredentialStore cmdlets
• Use a native encryption method
• POC with no encryption
Proxy Functions

In essence, a simple concept

- Generates a "wrapper" function around original cmdlet
- Intercept before the cmdlet call, or replace with cmdlet of your own
Porting
Scheduling

Windows Task Scheduler

Crontab

Single function to cover both
Porting
Invoke-VMScript

Doesn’t work with all Guest OS platforms
Limited coverage of interpreters
Size limitation due to encoding
Not ready for side-by-side

Roll your own
• GuestOperations
• VMware Tools
• Invoke-VMScriptPlus
Invoke-VMScriptPlus

Based on Guest Operations
Requires VMware Tools
Side-by-side possible
Multiple interpreters
Full scripts to Linux family guest OS
Environment variable support
Invoke-VMScriptPlus

$sInvoke = @{
    VM = $vm
    ScriptType = 'Bash'
    ScriptText = 'printenv'
    GuestUser = 'luc'
    GuestPassword = ConvertTo-SecureString -String 'VMware1!' -AsPlainText -Force
    GuestOSType = 'Linux'
    ScriptEnvironment = 'Msg1=Hello','Msg2=World'
}

Invoke-VMScriptPlus @sInvoke
New possibilities
InSpec
Turn security and compliance into code

Translate compliance into Code
Clearly express statements of policy
Move risk to build/test from runtime
Find issues early
Write code quickly
Run code anywhere
Inspect machines, data, and APIs

PART OF A PROCESS OF CONTINUOUS COMPLIANCE
Scan for Compliance → Build & Test Locally → Build & Test CI/CD → Remediates → Verify

https://www.chef.io/inspec/
control '2-disable-ssh' do
  title 'Disable SSH'
  desc 'Disable Secure Shell (SSH) for each ESXi host to prevent remote access to ESXi shell.'
  impact 1.0
  tsm_ssh_cmd = 'Get-Vmhost | Get-VMHostService | Where {$_._key -eq "TSM-SSH" -and $_._running -eq $False}'
  describe powercli_command(tsm_ssh_cmd) do
    its('exit_status') { should cmp 0 }
    its('stdout') { should_not be_empty }
  end
end

https://www.chef.io/inspec/
OpBot by OpVizor
Use PowerCLI to Monitor and manage your vSphere environment from Slack

http://try.opvizor.com/opbot

World’s 1st Slack Bot for VMware vSphere
You can easily monitor and manage your VMs with OpBot from anywhere using simple yet powerful commands. Supports Slack and VMware vSphere.

Download now and use it for FREE.
PowerCLI and Jenkins
Jenkins and PowerCLI

Automate your tests on vSphere and VMware Cloud on AWS

Use the existing docker image from https://hub.docker.com/r/matt9ucci/jenkins-powershell/

Run Command:


Modules are stored in Jenkins home folder, populate with PowerCLI on host machine using:

Save-Module -LiteralPath ~/jenkins/.local/share/powershell/Modules -Name VMware.PowerCLI -Repository PSGallery

1st time requires some setup but then the mounted volume will include the persistent data

No need for PowerShell Plugin, use the default shell plugin for jobs
Jenkins and PowerCLI

Run commands as a shell script:

```
pwsh -file '/var/jenkins_home/pcli/get-vm.psl'
```

See the list of available environment variables.
Jenkins and PowerCLI

Example Script

write-output "Running PowerCLI Script"

if (-not (Get-Module -Listavailable VMware.PowerCLI)){
    Write-Output "PowerCLI not found in users folder, run the following from the docker host where ~/Jenkins/ is the volume used on startup"
    Write-output " Save-Module -LiteralPath ~/jenkins/.local/share/powershell/Modules -Name VMware.PowerCLI -Repository PSGallery"
} Else {
    Set-PowerCLIConfiguration -InvalidCertificateAction Ignore -confirm:$false
    Connect-VIServer -Server vcenter.sddc-x-x-x-x.vmwarevmc.com -Protocol https -User cloudadmin@vmc.local -Password ‘vytxx99**6’
    Get-VM
}
Jenkins and PowerCLI

Use Cases

Test PowerCLI Scripts on a regular basis

Spin up test VMs and run tests then destroy

Invoke Guest Operations

Advanced:

• Spin up 1 node VMware Cloud on AWS SDDC
• Configure Content Library to sync with Local Library
• Deploy Test VMs from Content Library
• Run Test cases inside VMs
• Save results
• Destroy SDSDC

Save money with testing as a service on VMware Cloud on AWS
Get started with VMware Cloud on AWS now!

**Starter Single Host SDDC**

PROMO: 20% off for 3 months*

http://vmwa.re/vmc1host

**Production 3 Host SDDC**

PROMO: For price of 2 Hosts*

offer starts Sep 10, 2018 till Nov 2, 2018

http://vmwa.re/vmc3host

Please visit https://cloud.vmware.com/vmc-aws/pricing for promotion details

* Limited time Offer – T&C’s apply.
PowerCLI & Containers
Containers

Pros & Cons of Containers
They are cattle is my #1
Demos with Photon OS 2.0
...but any Container-capable OS should do
VMware/PowerCLICore
  • Run standalone
  • Docker compose
Photon – Docker - PowerCLI

docker run --rm -it -d --name PCLIBox vmware/powerclicore
docker exec -t PCLIBox pwsh -c Get-PowerCLIConfiguration

- Quick
- Single/few cmdlets
- Not for scripts
Photon – Docker - PowerCLI

cat > ~/scripts/test.ps1 << EOF
Set-PowerCLIConfiguration -InvalidCertificateAction Ignore Confirm: \$false | Out-Null
EOF
chmod +x ~/scripts/test.ps1
docker run --rm -it -d --name PCLIBox -v ~/scripts:/scripts vmware/powerclicore
docker exec -t PCLIBox pwsh -c '/scripts/test.ps1 > /scripts/report.txt'

• Escape special characters ==> error prone
• Chmod required
• Output in container
mkdir /mnt/SharedScripts

mount -t cifs //ws1.local.lab/vmworld2018 /SharedScripts -o user=lucd,password=$pswd
docker run --rm -it -d --name PCLIBox -v /SharedScripts:/scripts vmware/powerclicore
docker exec -t PCLIBox pwsh -c '/scripts/VIN1709BU/test.ps1'

umount /SharedScripts

• Cifs/Nfs share
• Edit with Visual Studio Code on your station – github repo
• Result on same share/separate share
Key Takeaways

1. PowerShell and PowerCLI is everywhere!
2. Open = New Possibilities
3. Remember the caveats
PLEASE FILL OUT YOUR SURVEY.

Take a survey and enter a drawing for a VMware company store gift card.
POSSIBLE
BEGINNS
WITH YOU

THANK YOU!