Welcome to the Synergy Roundtable

created by Stephan Koch & Dirk Derichsweiler -
do not hesitate to contact us: stephan.koch@hpe.com or dirk.derichsweiler@hpe.com

jupyter Notebook can be found: https://github.com/dderichswei/synergy (https://github.com/dderichswei/synergy)

requirements

- Anaconda (Python) recommended (for the Jupyter installation and pip)
- Jupyter-Powershell (https://github.com/vors/jupyter-powershell (https://github.com/vors/jupyter-powershell))

additional information

On your HPE OneView appliance, or online
https://10.0.20.50/help/cic-rest/en/content/index.html#home.html (https://10.0.20.50/help/cic-rest/en/content/index.html#home.html)
https://10.0.20.50/api-docs/current/ (https://10.0.20.50/api-docs/current/)
http://www.hpe.com/info/oneview/docs (http://www.hpe.com/info/oneview/docs)
https://developer.hpe.com/ (https://developer.hpe.com/)

Powershell (POSH) specific

https://github.com/HewlettPackard/POSH-HPOneView (https://github.com/HewlettPackard/POSH-HPOneView)

Installation of HP OneView Module

only required, if not installed yet.

Login

In [ ]:

```
#for mounting ISO via ILO
install-module -Name HPRESTCmdlets -Confirm:$false
get-command -module HPRESTCmdlets
```

import the PowerShell module:

In [ ]:

```
Import-Module -name hponeview.500
```

login:

In [ ]:

```
$username = "XXXXXXX"
$password = ConvertTo-SecureString "XXXXXXXX" -AsPlainText -Force
$psCred = New-Object System.Management.Automation.PSCredential -ArgumentList ($username
, $password)

Connect-HPOVMgmt -Hostname 10.0.20.50 -AuthLoginDomain local -Credential $psCred
```

In [ ]:

```
$token = $global:ConnectedSessions[0].SessionID
$token
```

ADVANCED: which functions are available?

In [ ]:

```
Get-Command -Module HPOneView.500
```

show existing networks

In [ ]:

```
Get-HPOVNetwork
```

create network

In [ ]:

```
New-HPOVNetwork -Name "Roundtable - Test Ethernet Network" -VlanId 200 -Type Ethernet -
VLANType Tagged -Purpose General -SmartLink $False -PrivateNetwork $False
```

create bulk network

How to setup multiple networks at once.

In [ ]:

```
$net = @(1,2,3,4,5,7,200)
foreach ($i in $net) { New-HPOVNetwork -Name "Bulk_$i" -VlanId $i -Type Ethernet -VLANT
ype Tagged }
```

delete bulk network

as it´s not needed for the demo

In [ ]:

```
Get-HPOVNetwork -name "bulk*" | Remove-HPOVNetwork -Confirm:$false
```

show configured/existing storage (systems and pools)

In [ ]:

```
Get-HPOVStorageSystem
```

In [ ]:

```
Get-HPOVStoragePool  | select Name, URI
```

show volume templates

In [ ]:

```
Get-HPOVStorageVolumeTemplate
```

create volume template

In [ ]:

```
$StorPool=Get-HPOVStoragePool -Name "SSD_r6" -StorageSystem primera630
New-HPOVStorageVolumeTemplate -Name "Roundtable Volume Template2" -StoragePool $StorPoo
l -Capacity 10240
```

show Enclosure / Server Hardware / Bay

In [ ]:

```
Get-HPOVEnclosureGroup
```

In [ ]:

```
Get-HPOVServerHardwareType
```

In [ ]:

```
Get-HPOVServer | select Name, URI
```

show Serverprofile

In [ ]:

```
Get-HPOVServerProfileTemplate|select Name, URI
```

In [ ]:

```
Get-HPOVServerProfile|select Name, URI
```

Deploy and Install new ESX Server

create server profile (takes 2-3 minutes)

In [ ]:

```
### Get the first available server based on the template configuration
##$Server = Get-HPOVServer -InputObject $ServerProfileTemplate -NoProfile | Select -Fir
st 1

$ServerName = "CTC H5 HE11, bay 1"
$Server=Get-HPOVServer -Name $ServerName

#Power off Server if on
Stop-HPOVServer -Server $Server -Force -Confirm:$false | Wait-HPOVTaskComplete

$Template=Get-HPOVServerProfileTemplate -Name "ANSIBLE_OS_Deploy_via_iLO"



$params = @{
        AssignmentType        = "Server";
        Description           = "HPE Synergy 480  Server ";
        Name                  = "Roundtable - API Demo Server (Stephan by POSH)";
        Server                = $Server;
        ServerProfileTemplate = $Template;
}

New-HPOVServerProfile @params | Wait-HPOVTaskComplete

#Power on Server
#Start-HPOVServer -Server $Server
```

create kickstart file

In [ ]:

```
#write ESX kickstart file on Webserver

$OSIP = "10.0.33.130"
$HOSTNAME = "esx01"
$OUT="/persistent/osdepl/esx67/ks_custom67.cfg"


'' | Out-File  $OUT
'# Sample scripted installation file' | Out-File -Append $OUT
'# Accept the VMware End User License Agreement' | Out-File -Append $OUT
'vmaccepteula' | Out-File -Append $OUT
'# Set the root password for the DCUI and ESXi Shell' | Out-File -Append $OUT
'rootpw HP1nvent!' | Out-File -Append $OUT
'# Install on the first local disk available on machine' | Out-File -Append $OUT
'clearpart --firstdisk --overwritevmfs' | Out-File -Append $OUT
'install --firstdisk=remote --overwritevmfs' | Out-File -Append $OUT
'# Set the network to DHCP on the first network adapater, use the specified hostname an
d # Create a portgroup for the VMs' | Out-File -Append $OUT
'network --bootproto=static --addvmportgroup=1 --ip=' + $OSIP + ' --netmask=255.255.25
5.0 --gateway=10.0.33.254 --nameserver=10.0.20.5 --hostname=' + $HOSTNAME + ' --device=
vmnic0' | Out-File -Append $OUT
'# reboots the host after the scripted installation is completed' | Out-File -Append $O
UT
'reboot' | Out-File -Append $OUT
'' | Out-File -Append $OUT
'%firstboot --interpreter=busybox' | Out-File -Append $OUT
'# Add an extra nic to vSwitch0 (vmnic2)' | Out-File -Append $OUT
'esxcli network vswitch standard uplink add --uplink-name=vmnic1 --vswitch-name=vSwitch
0' | Out-File -Append $OUT
'# Assign an IP-Address to the first VMkernel, this will be used for management' | Out-
File -Append $OUT
'# esxcli network ip interface ipv4 set --interface-name=vmk0 --type=dhcp' | Out-File -
Append $OUT
'# esxcli network vswitch standard portgroup add --portgroup-name=vMotion --vswitch-# n
ame=vSwitch0' | Out-File -Append $OUT
'esxcli network vswitch standard portgroup set --portgroup-name=vMotion' | Out-File -Ap
pend $OUT
'esxcli network ip interface add --interface-name=vmk1 --portgroup-name=vMotion' | Out-
File -Append $OUT
'# esxcli network ip interface ipv4 set --interface-name=vmk1 --type=dhcp' | Out-File -
Append $OUT
'# Enable vMotion on the newly created VMkernel vmk1' | Out-File -Append $OUT
'vim-cmd hostsvc/vmotion/vnic_set vmk1' | Out-File -Append $OUT
'# Add new vSwitch for VM traffic, assign uplinks, create a portgroup and assign a VLAN
ID' | Out-File -Append $OUT
'# esxcli network vswitch standard add --vswitch-name=vSwitch1' | Out-File -Append $OUT
'# esxcli network vswitch standard uplink add --uplink-name=vmnic1 --vswitch-name=vSwit
ch1' | Out-File -Append $OUT
'# esxcli network vswitch standard uplink add --uplink-name=vmnic3 --vswitch-name=vSwit
ch1' | Out-File -Append $OUT
'# esxcli network vswitch standard portgroup add --portgroup-name=Production --vswitch-
name=vSwitch1' | Out-File -Append $OUT
'# esxcli network vswitch standard portgroup set --portgroup-name=Production --vlan-id=
10' | Out-File -Append $OUT
'# Set DNS and hostname' | Out-File -Append $OUT
'# esxcli system hostname set --fqdn=esxi5.localdomain' | Out-File -Append $OUT
'esxcli network ip dns search add --domain=demo.local' | Out-File -Append $OUT
'esxcli network ip dns server add --server=10.0.20.5' | Out-File -Append $OUT
'esxcli network ip dns server add --server=10.0.20.6' | Out-File -Append $OUT
```

```
'# Set the default PSP for EMC V-MAX to Round Robin as that is our preferred load balan
cing mechanism' | Out-File -Append $OUT
'# esxcli storage nmp satp set --default-psp VMW_PSP_RR --satp VMW_SATP_SYMM' | Out-Fil
e -Append $OUT
'# Enable SSH and the ESXi Shell' | Out-File -Append $OUT
'vim-cmd hostsvc/enable_ssh' | Out-File -Append $OUT
'vim-cmd hostsvc/start_ssh' | Out-File -Append $OUT
'vim-cmd hostsvc/enable_esx_shell' | Out-File -Append $OUT
'vim-cmd hostsvc/start_esx_shell' | Out-File -Append $OUT
```

mount ISO on ilo, set "next boot from", power on server

In [ ]:

```
# mount ISO on ilo, set "next boot from", power on server

#$ServerName = "CTC H5 HE11, bay 2"
#$Server=Get-HPOVServer -Name $ServerName


$ILOIP = $Server.mpHostInfo.mpIpAddresses[1].address
$IsoUrl = "http://osdepl.demo.local/esx67/esx67u3custom.iso"

#ilo User defined in Server Profile (Template)
$User = "XXXXXX"
$PW = "XXXXXX"


# Creation of the header

$body1 = @{UserName=$User;Password=$PW} | ConvertTo-Json
$headers = @{}
$headers["Content-Type"] = "application/json"
$headers["OData-Version"] = "4.0"

$URL = "https://$ILOIP/redfish/v1/SessionService/Sessions/"
$response = Invoke-WebRequest $URL -SkipCertificateCheck -ContentType "application/jso
n" -Method 'POST' -Headers $headers -Body $body1
$Token = $response.Headers['X-Auth-Token']



$headers["Content-Type"] = "application/json"
$headers["X-Auth-Token"] = "$Token"
#$headers

#Eject Media
$URL = "https://$ILOIP/redfish/v1/Managers/1/VirtualMedia/2/Actions/VirtualMedia.EjectM
edia/"
$response = Invoke-WebRequest $URL  -SkipCertificateCheck -ContentType "application/jso
n" -Method 'POST' -Headers $headers
#$response.StatusDescription

#$body = @{Image= $IsoUrl;"Oem"= @{"Hpe"= @{"BootOnNextServerReset"= $True}}} | Convert
To-Json
$body = @{Image= $IsoUrl } | ConvertTo-Json


#Mount Media
$URL = "https://$ILOIP/redfish/v1/Managers/1/VirtualMedia/2/Actions/VirtualMedia.Insert
Media/"
$response = Invoke-WebRequest $URL  -SkipCertificateCheck -ContentType "application/jso
n" -Method 'POST' -Headers $headers -Body $body

$response.StatusDescription

# Patch BootOnNextServerReset= $True
$body = @{"Oem"= @{"Hpe"= @{"BootOnNextServerReset"= $True}}} | ConvertTo-Json
$URL = "https://$ILOIP/redfish/v1/Managers/1/VirtualMedia/2/"
$response = Invoke-WebRequest $URL  -SkipCertificateCheck -ContentType "application/jso
n" -Method 'PATCH' -Headers $headers -Body $body

$response.StatusDescription
```

```
#Power on Server
Start-HPOVServer -Server $Server
```

join new ESX Server into vcenter

In [ ]:

```
##  install-module VMware.PowerCLI  needed !

do {
  $ping = Test-Connection -TargetName $OSIP  -TcpPort 22
  write-host "waiting ..."
  sleep 5
} until ($ping)

write-host "installation finished ..."
# join vcenter

$myvcenter = Connect-VIServer -Server suo04ctcvcsa001.demo.local -Protocol https -User
XXXXX -Password XXXXXX -Force
Add-VMHost -Server "suo04ctcvcsa001.demo.local" -Name $OSIP -Location Democluster -User
root -Password HP1nvent! -Force
```

Preperation for ESX kickstart Installation

Get an ESX ISO image and mount it on an Linux Server

cp -pR it to an folder

Edit the boot.cfg under root of the CDROM and under the /efi/boot

```
add bootstate=0
title=Loading ESXi installer
timeout=5
prefix=
kernel=/b.b00
kernelopt=cdromBoot runweasel ks=http://osdepl.demo.local/esx67/ks_custom67.cfg
```

write an new customized iso with:

```
mkisofs -relaxed-filenames -J -R -b isolinux.bin -c boot.cat -no-emul-boot -boot
-load-size 4 -boot-info-table -eltorito-alt-boot -e efiboot.img -boot-load-size
 1 -no-emul-boot -o /tmp/customesxi.iso .
```

Place customized iso and kickstart file on an reachable web server

dhcsp server must be available

Place customized iso and kickstart file on an reachable web server

create an Server Profile Template

```
with an ilo user in in
with an network connection where the dhcp request could be handled
and in my case with an SAN disk and boot from SAN configuration
```

Backup

Redfish POSH Mount # valid certificates needed

In [ ]:

```
$ILOTOKEN=Get-HPOVIloSso -InputObject $Server
$ILOTOKEN
```

In [ ]:

```
#Example from: https://github.com/HewlettPackard/PowerShell-ProLiant-SDK/blob/master/HPERedfish/HPERedfishExamples.ps1
#These examples use HPE Redfish PowerShell cmdlets available at http://www.powershellgallery.com/packages/HPERedfishCmdlets/.
#These scripts provide examples of using HPE Redfish API on HPE iLO for common use cases.


function Set-VirutalMedia
{
    param
    (
        [System.String]
        $Address,

        [PSCredential]
        $Credential,

        [System.Object]
        $IsoUrl = $null,

        [System.Object]
        $BootOnNextReset = $null
    )

    Disable-HPERedfishCertificateAuthentication

    # NOTE: if ISO URL is blank and BootOnNextReset are blank/null, the virtual media is unmounted
    Write-Host 'Mount/Unmount virtual media DVD using URL'

    # Connect session
    $session = Connect-HPERedfish -Address $Address -Credential $Credential

    $managers = Get-HPERedfishDataRaw  -odataid '/redfish/v1/Managers/' -Session $session
    foreach($mgr in $managers.Members.'@odata.id')
    {

        $mgrData = Get-HPERedfishDataRaw -odataid $mgr -Session $session
        # Check if virtual media is supported
        if($mgrData.PSObject.Properties.name -Contains 'VirtualMedia' -eq $false)
        {
            # If virtual media is not present in links under manager details, print error
            Write-Host 'Virtual media not available in Manager links'
        }
        else
        {

            $vmOdataId = $mgrData.VirtualMedia.'@odata.id'
            $vmData = Get-HPERedfishDataRaw -odataid $vmOdataId -Session $session
            foreach($vm in $vmData.Members.'@odata.id')
            {
                $data = Get-HPERedfishDataRaw -odataid $vm -Session $session
                # select the media option which contains DVD
                if($data.MediaTypes -contains 'DVD')
                {
                    # Create object to PATCH to update ISO image URI and to set
```

```
                    if($IsoUrl -eq $null)
                    {
                        $mountSetting = @{'Image'=$null}
                    }
                    else
                    {
                        $mountSetting = @{'Image'=[System.Convert]::ToString($IsoUrl)}
                    }
                    if($BootOnNextReset -ne $null -and $IsoUrl -ne $null)
                    {
                        # Create object to PATCH
                        # for iLO 5
                        $oem = @{'Hpe'=@{'BootOnNextServerReset'=[System.Convert]::ToBo
olean($BootOnNextReset)}}

                        ## for iLO 4
                        #$oem = @{'Hp'=@{'BootOnNextServerReset'=[System.Convert]::ToBo
olean($BootOnNextReset)}}

                        $mountSetting.Add('Oem',$oem)
                    }
                    # PATCH the data to $vm odataid by using Set-HPERedfishData
                    #Disconnect-HPERedfish -Session $session
                    $ret = Set-HPERedfishData -odataid $vm -Setting $mountSetting -Sess
ion $session

                    # Process message(s) returned from Set-HPERedfishData
                    if($ret.error.'@Message.ExtendedInfo'.Count -gt 0)
                    {
                        foreach($msgID in $ret.error.'@Message.ExtendedInfo')
                        {
                            $status = Get-HPERedfishMessage -MessageID $msgID.MessageID
-MessageArg $msgID.MessageArgs -Session $session
                            $status
                        }
                    }
                    Get-HPERedfishDataRaw -odataid $vm -Session $session
                }
            }
        }
    }
    # Disconnect session after use
    Disconnect-HPERedfish -Session $session
}

 #Main
$ILOIP = "10.0.20.68"
$IsoUrl = "http://osdepl.demo.local/esx67/esx67u3custom.iso"

$User = "skoch"
$PWord = ConvertTo-SecureString -String "Passw0rd" -AsPlainText -Force
$cred = New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList $U
ser, $PWord

add-type @"
    using System.Net;
    using System.Security.Cryptography.X509Certificates;
    public class TrustAllCertsPolicy : ICertificatePolicy {
        public bool CheckValidationResult(
            ServicePoint srvPoint, X509Certificate certificate,
            WebRequest request, int certificateProblem) {
```

```
            return true;
        }
    }
"@
[System.Net.ServicePointManager]::CertificatePolicy = New-Object TrustAllCertsPolicy

[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Ssl3, [Net.Se
curityProtocolType]::Tls, [Net.SecurityProtocolType]::Tls11, [Net.SecurityProtocolType]
::Tls12 #Mount


Set-VirutalMedia -Address $ILOIP -Credential $cred -IsoUrl $IsoUrl -BootOnNextReset $tr
ue


#unmount
##Set-VirutalMedia -Address $Address -Credential $cred
```