Mike Nelson – SSV article

Quick Tips on Managing XenServer with PowerShell

PowerShell has been my admin tool of choice for a while now, making automated tasks much easier for me in my vSphere and Hyper-V environments. I also do a lot of Citrix work, but it always seemed to me that Citrix was very much behind in the adoption of PowerShell, and was, and still is in my opinion, lacking any solid support for what they have released for XenApp and XenDesktop. The XenServer snap-in, although still limited in the number of cmdlets available, is a tool no XenServer Administrator who uses PowerShell can be without, even if you are a religious XenClient or "xe" command line user. By adding this cmdlet arsenal to your existing Citrix PowerShell toolbox, you could manage your Citrix XenDesktop, XenApp, Provisioning Server, Netscaler, and XenServer environments all within a single CLI.

You can download the XenServer PowerShell snap-in from the Citrix SDK website. I would also recommend downloading the XenServer Snapshots snap-in created by Shannon Ma if you do use Snapshots in your environment. The XenServer snap-in is pretty easy to setup on a 32bit client OS. But if you are using Windows 7 or Server 2008R2 (yes, some folks do have that as their client OS), you will find it will not work. If you do a search for some help on this issue, you'll find a few folks that have posted the solution. However, it puzzles me to no end why Citrix has not officially published this anywhere. To get the snap-in working under a x64 system, you need to run the following command from an elevated command prompt to register the DLL with the 64bit .NET Framework –

```
C:\windows\microsoft.net\frameowkr64\v2.0.50727\installutil.exe "c:\program files (x86)\citrix\xenserverpssnapin\xenserverpssnapin.dll"
```

You should see a final "The commit phase completed successfully" message meaning it all registered fine. You then need to add it to your PowerShell session or Profile. I prefer to add it to my profile so it is always included. To check to see if it is already loaded and load it if necessary, add the following code to your PowerShell Profile –

```
if (((Get-PSSnapin -Name "XenServerPSSnapIn" -ErrorAction SilentlyContinue) -eq $null ) -and ((Get-PSSnapin –registered -Name "XenServerPSSnapIn") -ne $null))

{ Add-PSSnapin XenServerPSSnapIn

."C:\Program Files\Citrix\XenServerPSSnapIn\Initialize-Environment.ps1" }
```

Now that you've got it loaded, make sure to do a "Connect-Xenserver" to login to your host as the root user, and let's take a look at some useful cmdlets and scripts you can use to help you do your work.

We'll first go over some of the more basic commands and scripts you could use. As with all PowerShell modules and snap-ins, the *Get-Command* and *Get-Help* cmdlets are your best friends when you need to see syntax and examples. These are some examples in my toolbox that I use frequently:

- To create 3 guests based off a template, and specify their names (great for creating POC and lab environments)

    - *Copy-LocalVM name1 name2 name3 1 3*
- To see what templates are available to use
    - *Get-Template*
- Find out who the Master of the Pool is
    - *Get-XenServer:Pool.Master*
- To set a homeserver for a VM
    - *Set-XenServer:VM.Affinity –VM <name> -Affinity <xenserver_name>*
- Set the License Server for the host and the License Edition
    - *Set-XenServer:Host.LicenseServer –server <xenserver> -Host xenserver -LicenseServer <license_host>*
    - *Invole-XenServer:Host.ApplyEdition –server <xenserver_name> -Host xenserver –Edition <enterprise, advanced>*

By combining these commands and a few more into a single script, we could basically create our own automated build environment for XenServer hosts and guests. Cody Bunch contributed his "Demo 2 – VM Easy Bake" script to the Poshcode.org repository and it is a great example for you to use to start creating your own scripts to create, change, report on, and delete your XenServer hosts and VM guests. He has also contributed a few more which you can search for at Poshcode.org, including a "Hypervisor Independent" script that works with both XenServer and VMware.

One last really cool thing to show you is using PowerShell to sort and interpret data that is collected from the OS command line itself. One thing I always look at when investigating a XenServer environment, especially with VDI, is the Disk I/O usage. Fortunately, XenServer has the built-in Linux command *iostat* to show us the actual disk I/O for the host and the running guests over a period of time that we can specify (If you haven't used *iostat* yet, I suggest you go to the Linux MAN Pages to learn more about this awesome piece of code.) The problem comes in when you have all of this data from iostat that you need to decipher, as it isn't very good at formatting it's output data and simply sends the text to a text file that you would specify. The Virtualization Jedi has come up with an awesome script to read those text files and format them as space-delimited files that can opened in Excel (or Google Spreadsheets, if you prefer) and sorted and analyzed to your heart's content. You can find all of the information in this posting on his blog.

By having this new set of cmdlets added to your existing Citrix Admin PowerShell Toolbox, you could be administrating your entire environment via PowerShell. Join these with your Windows OS, Hyper-V, and VMware PowerShell toolboxes, and you can harness the power and simplicity of scripting your Administration procedures and tasks. In a future article, I plan on going over the use of the PowerShell cmdlets for the Netscaler product line as well, so stay tuned!