Technical Report

Windows Command-line Automation Techniques for Dell EqualLogic PS Series Arrays

Abstract

This Technical report will describe different methods of command line automation available for Windows and Dell EqualLogic PS Series arrays. Examples are shown with step by step building guides in order to help administrators create their own customizable scripts.

TR1089

V1.0

Authored By: Chuck Farah

WWW.DELL.COM/PSseries

Preface

PS Series arrays optimize resources by automating performance and network load balancing. Additionally, PS Series arrays offer all-inclusive array management software, host software, and free firmware updates.

Audience

The information in this guide is intended for storage administrators.

Related Documentation

For detailed information about PS Series arrays, groups, volumes, array software, and host software, log in to the Documentation page at the customer support site.

Dell Online Services

You can learn about Dell products and services using this procedure:

1. Visit http://www.dell.com or the URL specified in any Dell product information.

2. Use the locale menu or click on the link that specifies your country or region.

Dell EqualLogic Storage Solutions

To learn more about Dell EqualLogic products and new releases being planned, visit the Dell EqualLogic TechCenter site: http://delltechcenter.com/page/EqualLogic. Here you can also find articles, demos, online discussions, technical documentation, and more details about the benefits of our product family.

For an updated Dell EqualLogic compatibility list please visit the following URL: https://support.equallogic.com/compatibility

Table of Contents

Table of Contents iv
Revision Information v
Introduction 1
Executive Summary 1
Available Dell EqualLogic Automation from the command line 1
PS Series Storage automation: the Goal of this document 2
Windows PowerShell 2
PS Series Arrays PowerShell 2
    Other methods of automation available with Dell PS Series Storage (these methods will not be covered in detail in this document) 3
    PS Series Arrays Group Manager CLI 3
    PS Series Arrays Host Scripting Tools 3
    PS Series Arrays Auto-Snapshot Manager (ASM) CLI 3
PS Series Arrays SAN Headquarters CLI 3
Windows® and Dell™ PS Series Automation Techniques 4
    Windows PowerShell ISE Setup 4
    Windows PowerShell Dell EqualLogic PS Series Group Access 10
    Windows PowerShell Dell EqualLogic PS Series Volume Creation and Access Automation 16
    Windows PowerShell Dell EqualLogic PS Series Volume Deletion and iSCSI Connection removal 30
Windows Server 2012 Native Windows Storage Management Automation Techniques 39
    Add volumes example with native Windows Server 2012 cmdlets 39
    Removing volumes with native Windows Server 2012 cmdlets 45
Conclusion 48
Appendix A: Full Script to add a volume (Windows 2008) 49
Appendix B: Full Script to delete a volume (Windows 2008) 54
Appendix C: Troubleshooting Techniques 57
Appendix D: Windows 2008 Integrated Scripting Environment (ISE) setup 59
Technical Support and Customer Service 60

Revision Information

The following table describes the release history of this Technical Report.

Report	Date	Document Revision
1.0	June	Initial Release

The following table shows the software and firmware used for the preparation of this Technical Report.

Vendor	Model	Software Revision
Dell	Host Integration Tools for Microsoft, includes Auto-Snapshot Manager/ME, PowerShell tools for De	V4.5, V4.6
Dell	SAN Headquarters	V2.5, V2.6

The following table lists the documents referred to in this Technical Report. All PS Series Technical Reports are available on the Customer Support site at: *support.dell.com*

Vendor	Document Title

Introduction

Dell PS Series Arrays provide powerful command line integration for Windows, Linux and VMware hosts. Windows PowerShell is the robust scripting interface that will allow the administrator to customize, automate and optimize their Dell storage operations. Dell EqualLogic PS series arrays are designed to provide simplicity, advanced integration and automatic optimization to fully leverage the storage investment.

Dell EqualLogic Host Integration Tools are available for no additional cost and includes PowerShell Commands unique to the PS Series Arrays as well as storage management automation through native Windows PowerShell commands for Windows Server 2012 and Windows 8. This document will focus on the Windows interfaces to the PS Series Arrays.

Executive Summary

The goal of this document is to arm the administrators with skills to help automate Windows® environments attached to Dell PS Series arrays. Examples and step by step guides are offered in this document to be used for reference or training to help demonstrate scripting possibilities with PS Series PowerShell commands (also known as cmdlets). The document will describe the different forms of application interface communication available for PS Series arrays as well as some useful Windows specific commands where appropriate. All of the Dell PS Series software described in this document is included with the PS Series array.

Available Dell EqualLogic Automation from the command line

Dell PS Series array command line application interfaces available:

- PS Series Group Manager CLI
- Dell EqualLogic Host Scripting tools for Windows, Linux and Unix
- Dell™ EqualLogic Host Integration Tools for Windows® includes Auto-Snapshot Manager/Microsoft Edition (ASM/ME) – enabling the ability to create data- and application-consistent Smart Copies of NTFS volumes , and
    - Dell EqualLogic PowerShell cmdlets specific to the PS Series.
    - Auto-Snapshot Manager CLI
- SAN HQ management CLI
- Windows® Server 2012 native PowerShell cmdlets through integration with the Windows® storage architecture and Dell's PS Series Storage Management Provider (SMP).

In addition to the PS Series specific APIs native Windows utilities such as DISKPART or WMI provide the ability to match up operating system objects to the physical properties necessary to manage the Dell EqualLogic PS Series storage. For instance DISKPART is used to format and partition iSCSI connected volumes on PS Series arrays to enable the volume for use in Windows® environments.

PS Series Storage automation: the Goal of this document

This document will use several examples to demonstrate building blocks for advanced automation with Dell EqualLogic PS Series Windows PowerShell cmdlets, native Windows PowerShell cmdlets, Windows Management Instrumentation (WMI), DISKPART and the iSCSI command line interface (CLI) utility.

The Windows PowerShell ISE scripting environment will be used to provide examples which demonstrate integration with PS Series arrays. The following will be achieved in this document:

1. How to enable and configure the PowerShell environment for Windows
2. How to list PS Series Array properties such as volumes, members and group information from PS Series PowerShell cmdlets.
3. How to create, discover, login and initialize a volume to a Windows host
4. How to delete existing volumes from Windows.
5. Show Windows Server 2012 native PowerShell cmdlet integration
    a. Automate multiple volume creation
    b. Automate multiple volume deletion

Windows PowerShell

Windows PowerShell is quickly becoming the popular scripting tool for advanced automation. With Windows PowerShell an Integrated Scripting Environment (ISE) is available to enable rapid development and debugging of the code. The scripting environment includes small commands which perform specific operations known as command-lets (cmdlet for short). These cmdlets are the building blocks for the advanced functionality found in modern Windows scripts today.

Windows® Server 2012 and Windows® 8 have a Storage Management Architecture that allows for external storage to be managed by native Windows Applications such as File Services and Systems Center Virtual Machine Manager (SCVMM). The Storage Management API is available for Dell EqualLogic storage with the Host Integration Tools for Microsoft® v4.5 (or higher) also known as the HIT for Microsoft. Host Integration Tools for Microsoft® v4.6 will need to be installed for full support of SCVMM 2012 SP1.

PS Series Arrays PowerShell

Dell™ EqualLogic Host Integration Tools (HIT) for Microsoft® also includes a suite of Windows PowerShell cmdlets specific to PS Series arrays. The PowerShell commands included with the HIT for Microsoft adds specific PS Series Array management functionality and automation to the Windows PowerShell environment. The PS Series PowerShell commands include advanced management and discovery of the PS series Group, Member or Volume and iSCSI Sessions, Replication and Snapshots.

Other methods of automation available with Dell PS Series Storage (these methods will not be covered in detail in this document)

PS Series Arrays Group Manager CLI

The PS Series array may be configured or viewed through a standard Telnet/SSH session to perform diagnostics, initial group configuration or management. The PS Series Group Manager CLI provides typical support call initiated commands or initial setup of the array, however further administrative tasks may be automated with the Host Scripting Tools described below. For further information please download the "Group Manager CLI Reference Guide" from the PS Series Firmware download section of https://eqlsupport.dell.com

PS Series Arrays Host Scripting Tools

Host Scripting tools are available to issue PS Series Group Manager command line interface (CLI) commands from a remote host and capture the output of the commands for further action. The Host Scripting tools are available for Windows, Linux and UNIX. These host scripting tools allow for manipulation of the PS Series Group Manager CLI with Perl or Python scripting languages. For further information please download the "Host Scripting Tools" from the PS Series Firmware download section of https://eqlsupport.dell.com

PS Series Arrays Auto-Snapshot Manager (ASM) CLI

Dell™ EqualLogic Host Integration Tools for Windows® with Auto-Snapshot Manager/Microsoft Edition (ASM/ME) may be installed to provide full application consistent snapshot capabilities. Although the ASM GUI interface provides advanced automation functionality the ASM CLI provides an application interface to the PS Series array for additional customization of **Smart Copy™** management.

Note: For Windows **Smart Copy™** uses the PS Series arrays protection capability to create application consistent snapshots, clones and replicas to provide protection and recovery of important information assets.

For further information please download the "Auto-Snapshot Manager Microsoft® Edition – User's Guide" from the Host Integration Tools for Microsoft download section of https://eqlsupport.dell.com

PS Series Arrays SAN Headquarters CLI

SAN Headquarters (SAN HQ) is an advanced performance, configuration and alert monitoring system for PS Series arrays. Several tasks may be accomplished with the SAN HQ cli such as report automation, adding PS Series Groups, creating archives and exports as well as the ability to launch SAN HQ GUI with specific parameters.

For further information please download the "Installation and User's Guide" from the SAN Headquarters download section of https://eqlsupport.dell.com

Windows® and Dell™ PS Series Automation Techniques

Windows PowerShell ISE Setup

The PowerShell Integrated Scripting Environment (ISE) will be used to enable development, testing and debugging of PowerShell scripts or modules to configure Dell EqualLogic Storage to Windows. Alternatively the Dell PS Series PowerShell Tools interface may be used, however this document will focus on the use of the Windows Integrated Scripting Environment (ISE) which is included with Windows® 2008 R2 or higher (Please see Appendix D for more information). All the examples below use PowerShell v2.0 available with Windows® 2008 R2 SP1 and PowerShell v3.0 for Windows® Server 2012 and Windows® 8.

Download and install the Host Integration Tool for Microsoft® v4.5 or higher from https://eqlsupport.dell.com. The kit will include the Dell EqualLogic PowerShell Tools Reference Guide v4.5 which will provide detailed information about each cmdlet.

Pre-requisites:

- Windows PowerShell Integrated Scripting Environment (ISE)
  Please see Appendix D for feature information on Windows 2008.

- Host Integration Tools for Microsoft®
  Please refer to the Dell EqualLogic PowerShell Tools Reference Guide
  https://eqlsupport.dell.com

Warning:

The examples that follow will make changes to your systems and should be thoroughly tested and modified in a development environment prior to moving to production. Please use the examples as a reference and be aware that these scripts are provided as is without warranty of any kind. The PS Series PowerShell cmdlet library are fully supported through standard Dell agreements.

To setup up our PowerShell Environment the following will need to be performed

1. As indicated above download and Install Dell EqualLogic Host Integration Tools for Microsoft (HIT/Microsoft v4.5 or higher)

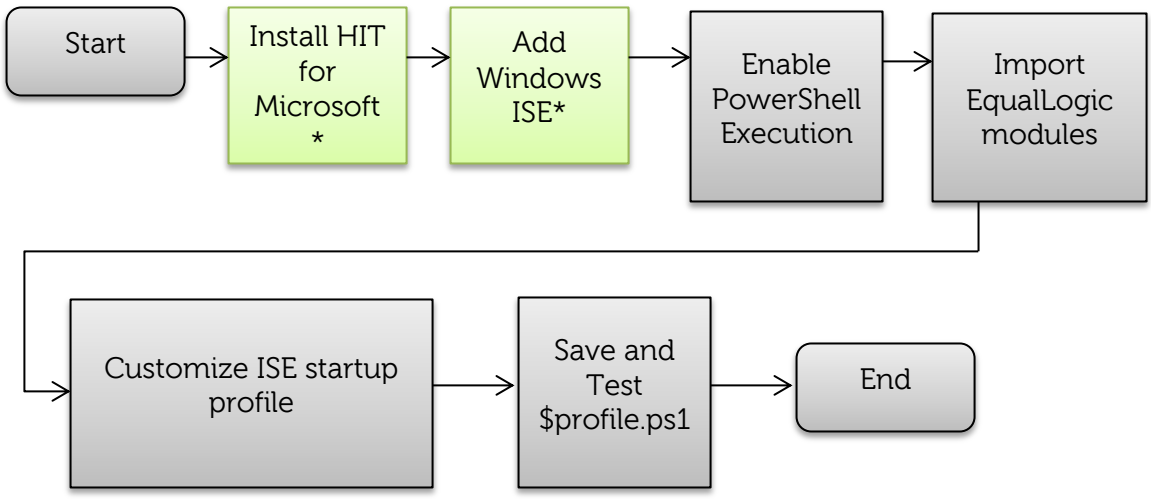
2. Add the Windows PowerShell ISE feature on Windows 2008 hosts
   Please see Appendix D for more information.

3. Enable Microsoft® Windows PowerShell script execution

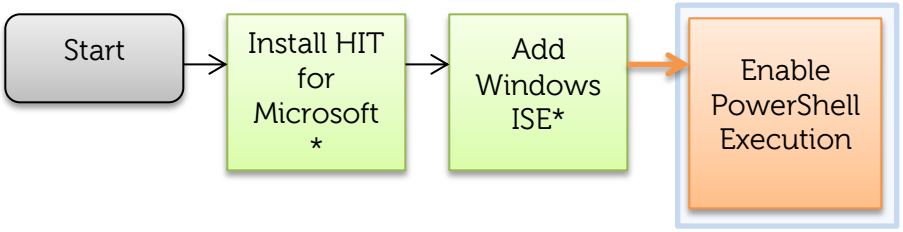
4. Customize the profile for Windows PowerShell ISE

5. Include the Dell EqualLogic PowerShell cmdlets module in the Windows PowerShell ISE

Graphical representation of the steps involved. The green will indicate tasks that are completed while the highlighted square will indicate the current process.



Step – By – Step Setup for Windows 2008 and Windows 2012/Windows 8



Note: HIT/Microsoft and Windows PowerShell ISE should be setup prior to this step in the instructions

1. Enable Windows PowerShell execution.
   By default the execution policy is set to "Restricted" which means you will not be able to execute any PowerShell Scripts.

   a. Open a Windows PowerShell console
   Windows Sever 2012\|Windows 8
   Server Manager->Tools->Windows PowerShell

   Windows 2008/Windows 7
   All Programs->Accessories->Windows PowerShell-> Windows PowerShell
   b. Determine execution policy: Enter at the command line

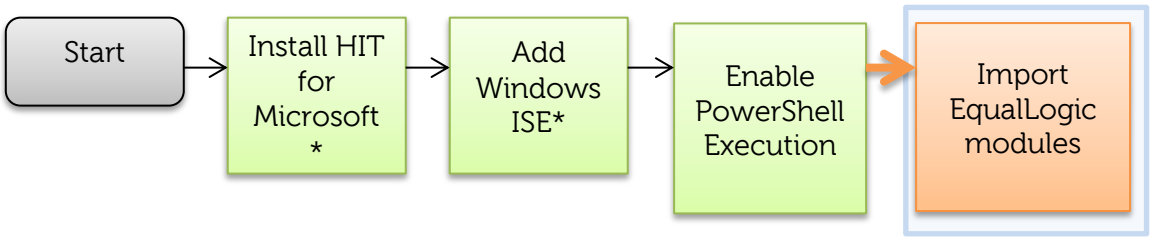
   ```
   PS C:\> Get-ExecutionPolicy
   ```

   c. If the execution policy is "Restricted" then enter :

   ```
   PS C:\> Set-ExecutionPolicy RemoteSigned
   ```

2. Setup the Windows ISE for PS Series cmdlets module.
   a. Open Windows ISE
   Windows Server 2012/Windows 8
   Server Manager->Tools->Windows PowerShell ISE

   Windows 2008/Windows 7
   All Programs->Accessories->Windows PowerShell->Windows PowerShell ISE
   (Right mouse – Run as administrator)



b. Import the PS tools (PowerShell cmdlets) module per session:

```
PS C:\> import-module –name <EqualLogic HIT Install
path>\EqlPSTools.dll

Example: import-module –name “C:\Program
Files\EqualLogic\bin\EqlPSTools.dll”
```

Test the access and list PS Series cmdlets:

```
PS C:\> Get-Command –Module EqlPSTools
```

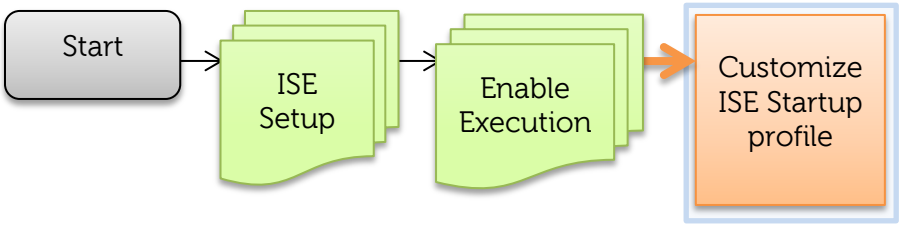
Example partial output:

```
PS C:\ > Get-Command -module EqlPSTools

CommandType     Name                                               ModuleName
-----------     ----                                               ----------
Cmdlet          Connect-EqlGroup                                   EqlPSTools
Cmdlet          ConvertFrom-EqlTemplateVolume                      EqlPSTools
```

Below are the guidelines to import the PS Tools (PowerShell cmdlets) permanently and provide persistence between Windows PowerShell ISE sessions.

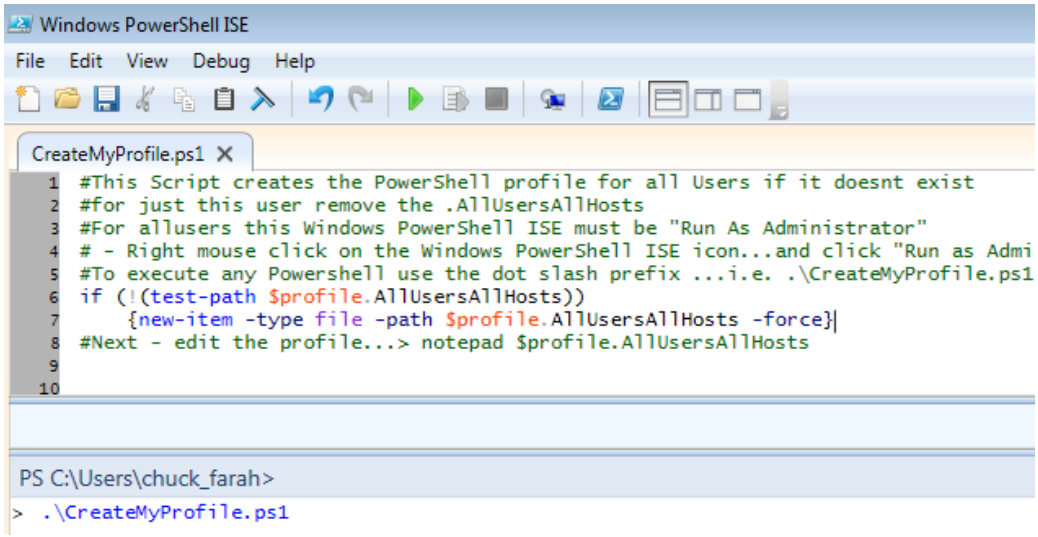
c. Create a profile or Open the appropriate profile script: for our example we will create a profile for all users and all hosts on this machine

d. Edit the profile to include an environmental variable to the path and the import the EqlPSTools module.



e. Create Profile:
Copy the code below to a new PowerShell document and save as **CreateMyProfile.ps1**

```
#########################
if (!(test-path $profile.AllUsersAllHosts))
    {new-item -type file -path $profile.AllUsersAllHosts -force}
```



f. Execute the PowerShell script from the command prompt:

```
PS C:> .\CreateMyProfile.ps1
```

Note: The ".\" (dot slash) prefix is a Microsoft security measure to ensure the authorized user is running the script.

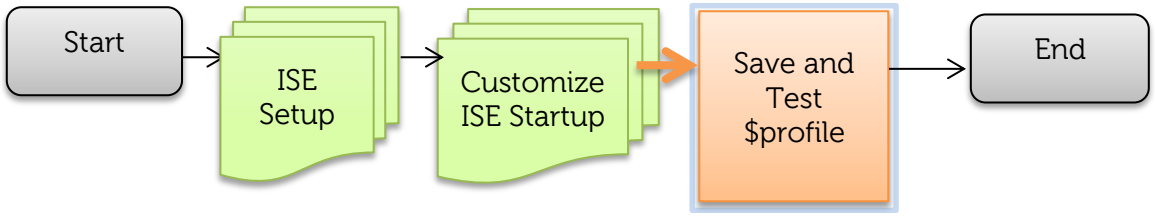
g. Edit the profile to add the module:

```
PS C:> notepad $profile.AllUsersAllHosts
```

h. Add these lines with the appropriate locations for the install path:

```
$env:PSModulePath = $env:PSModulePath + ";C:\Program
Files\EqualLogic\bin\EqlPSTools.dll"

Import-Module –name "C:\Program Files\EqualLogic\bin\EqlPSTools.dll"
```

Note: Be sure the install path is correct. Further customization is also available such as "CD" to a default specific working directory.



i. Save the profile, Exit the ISE and restart.

j. Once the Windows ISE is re-started test with:

```
PS C:\> Get-Command –Module EqlPSTools
```

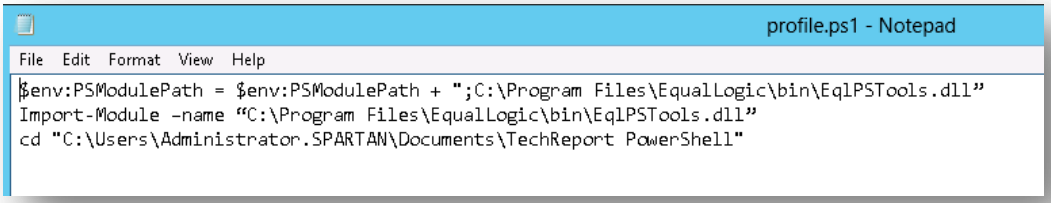
```
CommandType     Name                                               ModuleName
-----------     ----                                               ----------
Cmdlet          Connect-EqlGroup                                   EqlPSTools
Cmdlet          ConvertFrom-EqlTemplateVolume                      EqlPSTools
```

Note: These guidelines will import the EqualLogic PS Tools (PowerShell cmdlets) permanently and provide persistence between Windows PowerShell ISE sessions.

See: http://msdn.microsoft.com/en-us/library/dd878326(v=vs.85).aspx for more information.

Tip: Be sure the install path is correct. Further customization is also available such as "CD" to a default specific working directory.

Example of a Profile Script:



Windows PowerShell Dell EqualLogic PS Series Group Access

Before being able to execute PowerShell cmdlets that will manage PS Series storage the PS Series Group(s) will need to be registered and connected. This may be achieved through Auto-Snapshot Manager/Microsoft Edition or through the provided PowerShell cmdlets. We will use the PowerShell cmdlets option for this example. See the "Dell EqualLogic PowerShell Tools v4.5 Quick Reference" for more information.

The Following steps will be performed:

- Check for previous connection to the PS Series Groups
- Establish new group access if needed
- Connect to the PS series groups
- Test Connection

Step – By – Step PS Series Group Access

1. Start the Windows PowerShell ISE
   Windows Server 2012/Windows 8
   Server Manager->Tools->Windows PowerShell ISE

   Windows 2008/ Windows 7
   All Programs->Accessories->Windows PowerShell->Windows PowerShell ISE

2. Check for previous PS Series Group access :
   Windows Server 2012

   ```
   PS C:\> Get-StorageSubSystem
   ```

   This command uses the SMP integration in Windows Server 2012.

   ```
   PS C:\ > Get-StorageSubSystem
   WARNING: No EqualLogic PS Groups configured on this host. Please use EqualLogic
   PowerShell Tools or Auto-Snapshot Manager to configure PS Group access.
   ```

   Windows 2008/ Windows 7

   ```
   PS C:\> Get-EqlGroupAccess
   ```

   ```
   PS C:\> Get-EqlGroupAccess
   WARNING: No connection exists to any Group.
   ```

   Note: Group access may have been established during the installation of the Host Integration Tools for Microsoft. Multiple Groups may be registered and connected to a single host.

   All of the commands listed for Windows 2008 will work as well for Windows Server 2012.

3. Register PS Series Group access (not necessary if already configured but will need to verify connection in step 5.).

   Syntax:

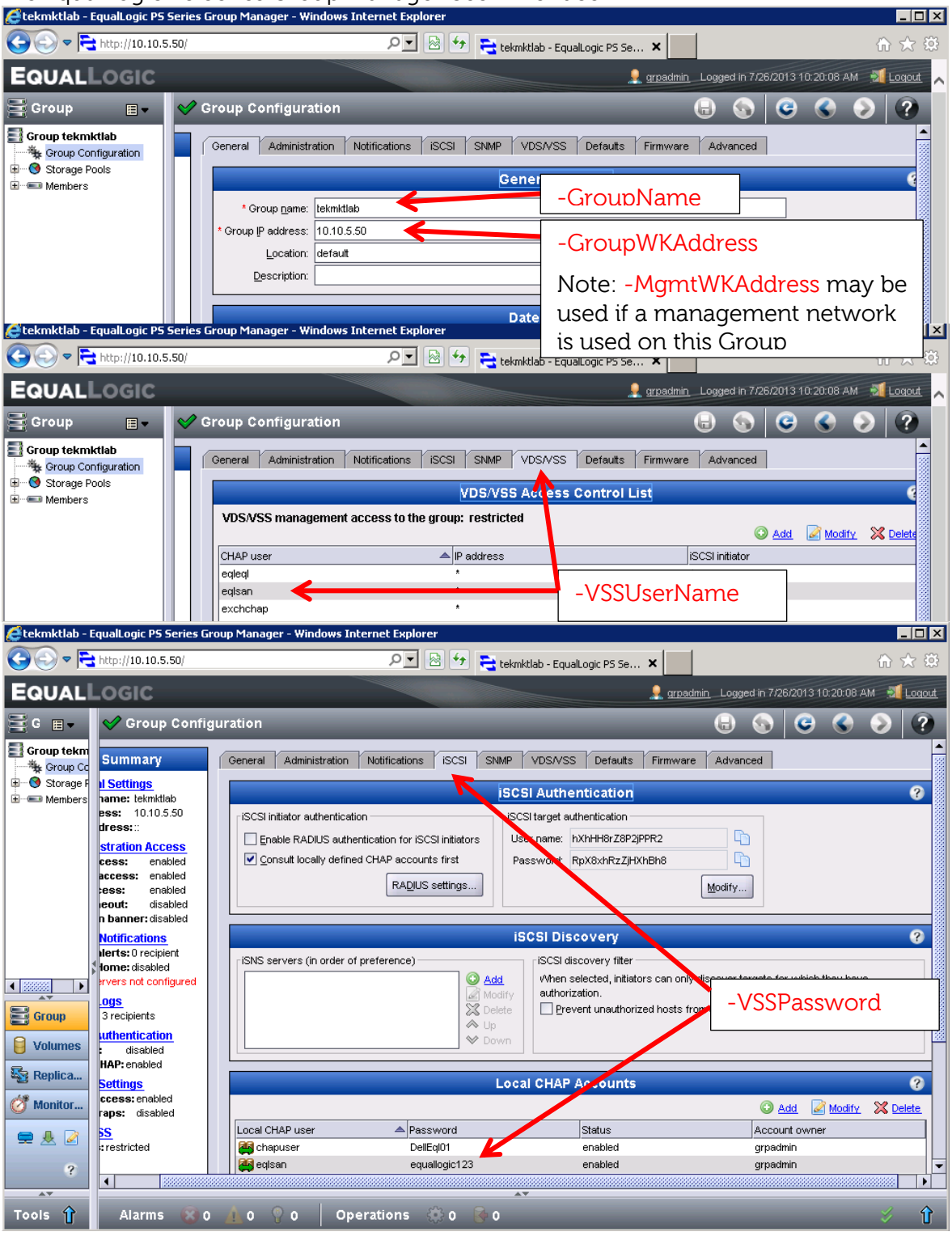
   ```
   New-EqlGroupAccess -Groupname <groupname> -GroupWKaddress
   <WellKnownIPaddress> -username <DOMAIN>\<username> -password <password> -
   VSSUserName <chapuserforvss> VSSPassword <chappassword>
   ```

   Example:

   ```
   PS C:\> New-EqlGroupAccess -GroupName tekmktlab -GroupWKAddress 10.10.5.50 -
   Username grpadmin -Password eqleql -VSSUsername eqlsan -VSSPassword equallogic123
   Successfully connected to the group 'tekmktlab'
   ```

   Note: New-EqlGroupAccess will register the array and connect. This is the preferred method for connecting to the array as opposed to Connect-EqlGroup.

Below screen captures will indicate how to locate the input parameters from the EqualLogic PS Series Group Manager User Interface



4. Check for all PS Series access and Connection information

```
PS C:\Get-EqlGroupAccess|Format-Table –property GroupName, SessionID
```

If the SessionID does not show as <disconnected> please continue to Step 6.

```
PS C:\> Get-EqlGroupAccess |format-Table -property groupname,sessionid -AutoSize

GroupName  SessionId
---------  ---------
TR2ProdGrp 64ED2A45484434AA50FB440000009035
```

Note: a SessionId populated indicates a successful connection to the Group. The Format-Table cmdlet allows for column selection and output format customization.

Multiple Groups may be connected to this Server (see example below):

```
PS C:\> Get-EqlGroupAccess|Format-Table -Property GroupName,SessionID -Autosize

GroupName      SessionId
---------      ---------
ArmyGroupT1    60006585C852CD2A141FE50100002000
tekmktlab      6090A02860F2A2B53C6F1483020020C7
tekmktlab-10Gb 6090A09840D4057EA9D5A4000000D086
```

To select just an individual Group use the following cmdlet:

```
PS C:\> Get-EqlGroupAccess|Where-Object  {$_.GroupName -eq 'tekmktlab'}|Format-
Table -Property GroupName,SessionID -Autosize

GroupName SessionId
--------- ---------
tekmktlab 6090A02860F2A2B53C6F1483020020C7
```

The Where-Object cmdlet is a handy way to filter selections.

5. *** Only Connect if the Group is currently disconnected ***

```
PS C:\Get-EqlGroupAccess|Format-Table –property GroupName, SessionID
```

```
PS C:\> get-eqlgroupaccess |ft -property groupname,sessionid -AutoSize

GroupName  SessionId
---------  ---------
TR2ProdGrp <disconnected>
```

If <disconnected> Connect to the PS Series array

```
PS C:\> Set-EqlGroupAccess -GroupName TR2ProdGrp -UserName grpadmin `
-Password eqleql
```

```
Successfully connected to the group 'TR2ProdGrp'
```

The success statement indicates that the credentials and connection information is correct. The Set-EqlGroupAccess is the preferred method to re-establish connections if needed.

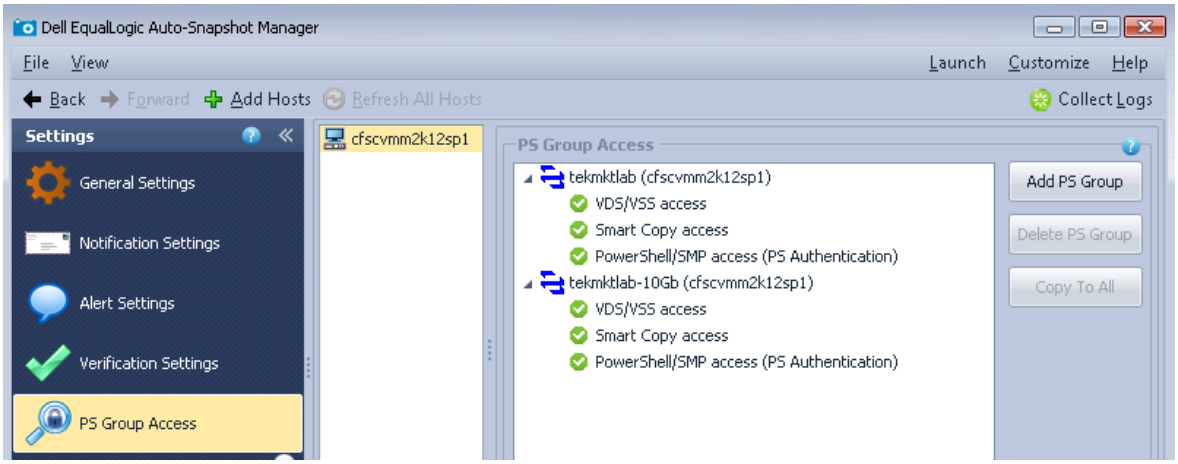
Note: In some cases you may need to re-register with the "Remove-EqlGroupAccess –GroupName $GroupName" and then "New-EqlGroupAccess".

6. To test connection and access to the Group:

```
PS C:\> Get-EqlGroup
```

```
SessionId                          : 6090A09840D4057EA9D5A4000000D086
groupId                            : 6090A09840D4057EA9D5A4000000D086
GroupName                          : tekmktlab-10Gb
GroupAddress                       : 10.10.6.50
GroupAddressIPv4                   : 10.10.6.50
GroupAddressIPv6                   : 0000:0000:0000:0000:0000:0000:0000:0000
ClientAccessSSH                    : enabled
ClientAccessTelnet                 : enabled
WebAccess                          : enabled
WebAccessNoEncrypt                 : enabled
ConnectionBalancing                : enabled
PerformanceBalancing               : enabled
GroupDate                          : 06/11/13
GroupTime                          : 10:22:44
GroupTimeZone                      : america_New_York
GroupDescription                   : 10 Gigabit Storage Group
UseCHAPForDiscovery                : disabled
EmailHomeContact                   :
SMTPServers                        : 10.124.4.210
EmailSender                        : joe@dell.com
EmailSenderDomain                  : dell.com
EmailSenderName                    : joe
EmailRecipientList                 : joe@dell.com;
EmailNotification                  : disabled
FTPService                         : enabled
UseLocalChapForIscsi               : enabled
GroupLocation                      : TechMkt R1
MembershipPassword                 : ****************
SyslogNotification                 : enabled
NTPServers                         : 10.124.4.103
SyslogServers                      : 10.124.4.210;10.17.225.67;172.16.213.184
RadiusIscsiAuthentication          : disabled
RadiusLoginAuthentication          : disabled
RadiusAuthenticationServers        :
RadiusAuthenticationSecrets        :
RadiusAuthenticationTimeoutSeconds : 2
RadiusAuthenticationRetryCount     : 1
RadiusUserAccounting               : disabled
RadiusAccountingServers            :
RadiusAccountingSecrets            :
RadiusAccountingTimeoutSeconds     : 2
RadiusAccountingRetryCount         : 1
LDAPLoginAuthentication            : disabled
OptimizedMPIOSessions              : Enabled
UnlimitedMPIOSessions              : Disabled
MinimalMPIOSessions                : Disabled
```

Note: This connection may also be shown with the native Windows Server 2012 PowerShell cmdlets “Get-StorageSubsystem” as well as the Auto-SnapShot/ME GUI.



Note: Auto-Snapshot Manager/Microsoft® Edition PS Group Access example.

Tip: Any command may be explained with Get-Help cmdlet <options> where options may be –detailed, -examples etc.

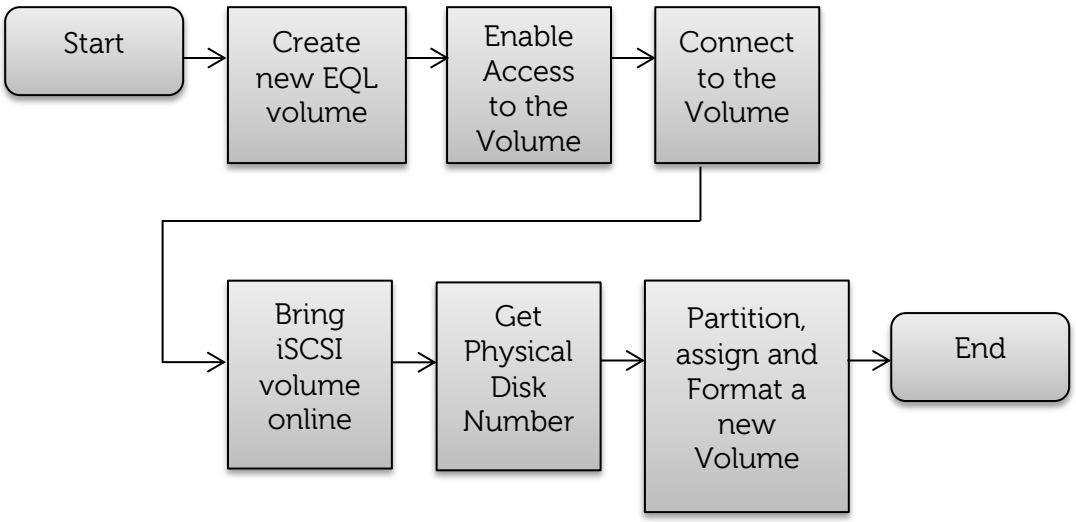
Windows PowerShell Dell EqualLogic PS Series Volume Creation and Access Automation

In this section we will demonstrate how to create a new volume, enable access and connect an iSCSI volume as well as format, partition and bring the volume online to Windows. This is the method required for Windows operating systems prior to Windows Server 2012 or Windows 8. Windows Server 2012 and Windows 8 include additional integration and capabilities with Dell EqualLogic storage. However the below scripts will work with Windows Server 2012, Windows 8 and Windows 2008.

The following example will use the Dell™ PS Series PowerShell cmdlets, Windows native PowerShell cmdlets, Windows iSCSI CLI (a Windows utility to manage iSCSI sessions), Windows Management Interface (WMI) as well as the DISKPART utility for Windows.

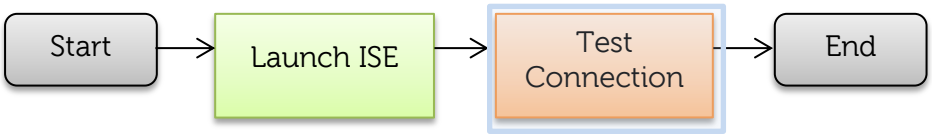
To add an EqualLogic volume through a command interface the following steps will be executed.

1. Create the new volume on the PS Series Array through the PS Series PowerShell cmdlets.
2. Enable the Access Control List (ACL) for that volume to the host initiator (to provide access to the volume through the iSCSI network).
3. Connect the volume created on the PS Series array to the host with the Windows iSCSI CLI (iscsicli.exe).
4. Determine the Physical Disk just created (from the WMI interface for Windows 2008 R2)
5. Bring the new volume online with the Windows DISKPART utility (Windows 2008) which will format a new NTFS volume assigned to the next available drive letter.



Step – By – Step Pre-Windows Server 2012

We will test each of these steps individually then provide a complete PowerShell script for latter modification and automation. See Appendix A for the full script.

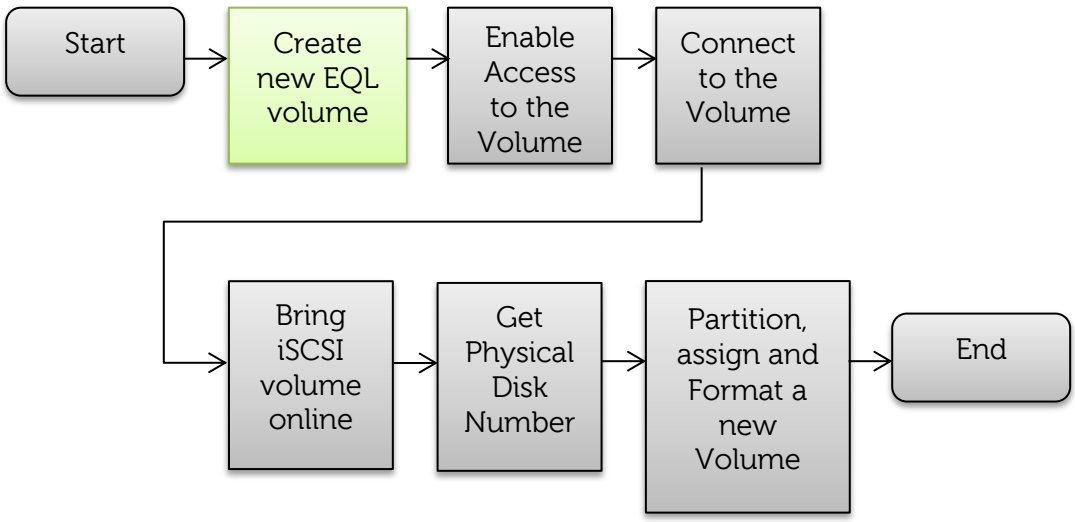


To complete this exercise please perform the following:

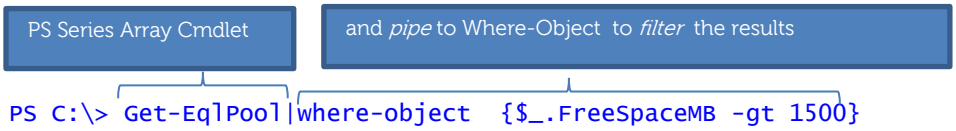
1. Launch the PowerShell ISE (**Note:** if you followed the steps to initialize your PowerShell environment you will not need to import the PS Series cmdlets)
2. Connect to the appropriate PS Series Group if needed. See the PS Series Group access section.
   To test connection and access to the Group:

```
PS C:\> Get-EqlGroupAccess|Format-Table –Property GroupName, SessionID
```

Note: If this command does not produce the connected PS Series arrays please follow the PS Series Group access instructions. In some cases the PS Group may need to re-register with the "Remove-EqlGroupAccess –GroupName $GroupName" and then "New-EqlGroupAccess".



3. Check to see if enough free space is available on the Pool for the volume. The following will list all Pools with 15GB or more free.

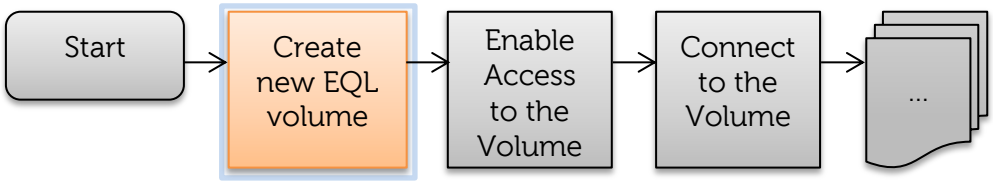


In this example the "default" pool is listed.

```
PS C:\> Get-EqlPool|Where-Object  {$_.FreeSpaceMB -gt 1500}


GroupId                          : 6090A09840D4057EA9D5A4
GroupName                        : tekmktlab-10Gb
GroupAddress                     : 10.124.2.30
StoragePoolName                  : default
PoolGuid                         : 08A09006-0900-5D44-004E-40B830BE1F00
Description                      :
IsDefaultPool                    : yes
PoolBorrow                       : Enabled
OnlineMemberCount                : 3
InUseMemberCount                 : 3
TotalSpaceMB                     : 64421865
FreeSpaceMB                      : 24537405
UsedSpaceMB                      : 24899475
ThinProvisionFreeSpaceMB         : 245374050
ISCSIConnectionCount             : 397
PoolActualFreeSpaceMB            : 24537405
PoolAvailableBorrowSpaceMB       : 29754795
PoolRecoverableSpaceMB           : 2625
PoolTotalBorrowSpaceMB           : 2625
PoolFreeBorrowSpaceMB            : 0
VolumeSubscribedMB               : 45747825
VolumeAllocatedMB                : 12639375
VolumeCount                      : 122
OnlineVolumeCount                : 117
InUseVolumeCount                 : 34
SnapshotReservedMB               : 6109830
SnapshotReservedAvailableMB      : 6109830
SnapshotSpaceUsedMB              : 275385
SnapshotCount                    : 48
OnlineSnapshotCount              : 12
InUseSnapshotCount               : 0
PoolSnapshotReserveBorrowSpaceMB : 2625
PoolSnapshotReserveFreeSpaceMB   : 6109830
PoolSnapshotTotalBorrowSpaceMB   : 0
```

If your Pool name is different simply change by entering:

```
PS C:\> $PoolName = 'myPoolName'
```



4. Initialize variables for your environment within the ISE command prompt (for ease of testing)

   Hint: Change the values below appropriately and copy then paste into the command input line in Windows PowerShell ISE.

   Copy/Paste

```
$MinFree = 15*1024 #~15GB free space avail
$VolUsable = 10*1024 #size of volume
$PoolName = 'MyPool' #NEED TO change to your pool
$GroupName= 'MyGroup' # USE your PS Series Group Name.
$GroupLabel='GrpLabel' #Reference to PS Group for label on the volume
#Example group label "PSPROD"
$ThisVol = 'MyVolume' #This is the volume name (change appropriately)
$TargetPortal= '127.0.0.1' #PS Series Group IP address
```

   Note: PS Series group IP may be located under "Discovery" in the iSCSI Initiator GUI.

   Each variable assignment may be typed individually or paste all the commands into the PowerShell command prompt as demonstrated below:

```
PS C:\> $MinFree = 15*1024 #15GB free space avail
$VolUsable = 10*1024 #size of volume
$PoolName = 'yourpool' #NEED TO change to your pool
$GroupName= 'yourgroup' # USE your PS Series Group ID.
$GroupLabel='TM10GB' #Reference to PS Group for label on the volume
$ThisVol = 'MyTestVol' #This is the volume name (change appropriately)
$TargetPortal= '127.0.0.0' #Enter you PS Series Group IP address discovery
tab of iscsi initiator tool
```

   Once you hit "enter" these variables will be initialized for this PowerShell ISE Session.

5. Create the new volume on the PS Series Array with usable of 10GB, Thin Provisioned and SnapShot Borrowing enabled.

   PS Series Array Cmdlet

```
PS C:\> New-EqlVolume -VolumeName $ThisVol -VolumeSizeMB $VolUsable -
ThinProvision yes -StoragePoolName $PoolName -SnapshotBorrowingEnabled
true -GroupName $GroupName
```

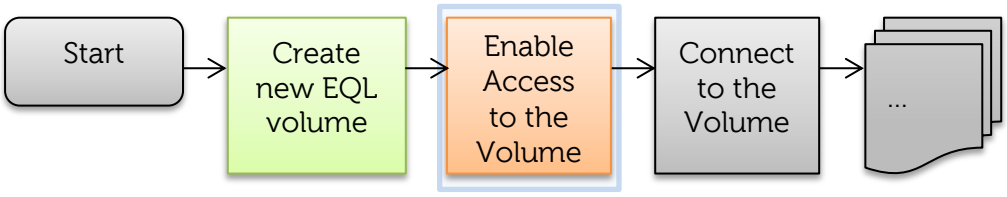
   Example output:

```
Volume 'TD1089V1' created successfully
```

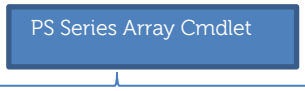
6. Determine the Host iSCSI Qualified Name (IQN) for Assess Control List (ACL) association

```
#Set the hostn variable to this host -using the environmental variable
PS C:\> $hostn = ($env:computername) #This Server or Host

#Set the object variable to the iscsi class for this host- using WMI
PS C:\> $object   = Get-WMiObject -namespace root\WMI -class
MSiSCSIInitiator_MethodClass -Computer $hostn

#Set the IQN (iSCSI Qualified Name) to the iSCSI node name from the WMI
Class
PS C:\ > $IQN = $object.iSCSINodeName
```

No output from these commands should occur. You may interrogate any variable simply by entering the variable at the command prompt. For instance: PS C:\ > $IQN <enter>#this will display your IQN.



Enable Access from this PS Series volume to this host



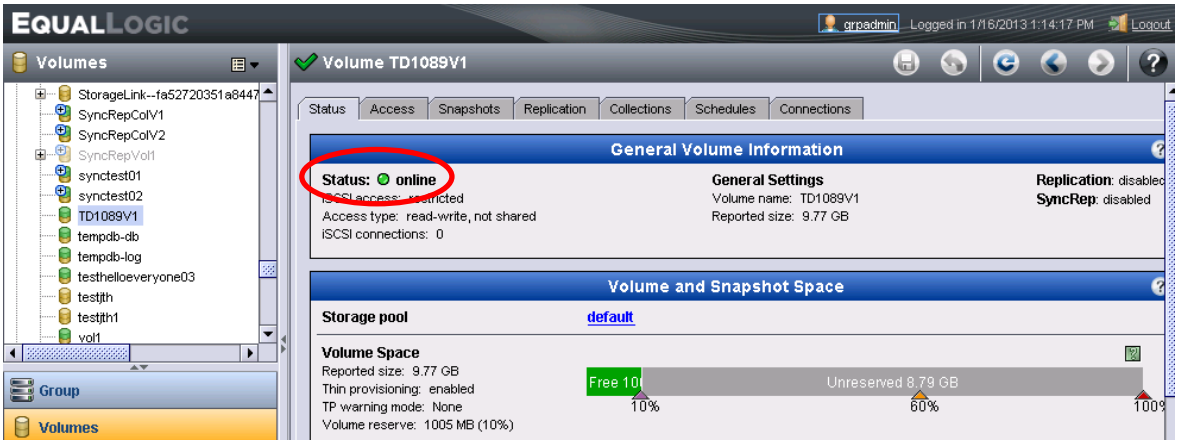

```
PS C:\> New-EqlVolumeAcl -VolumeName $ThisVol -InitiatorName $IQN -
AclTargetType volume_and_snapshot
```

Example Output:

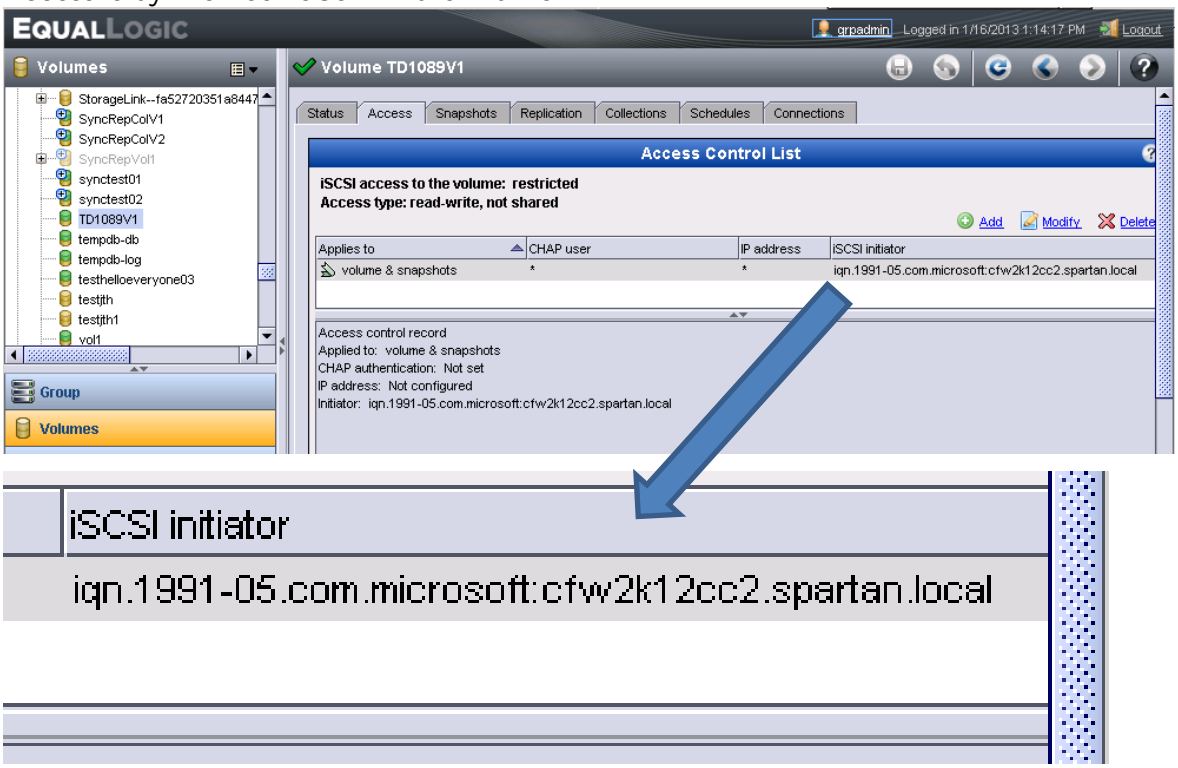
```
PS C:\> new-EqlVolumeAcl -volumename $ThisVol -InitiatorName $IQN -AclTargetType
volume_and_snapshot

PSAPI.Cmdlets.NewVolumeAcl TD1089V1 completed successfully.
```

7. Verify the status of the volume just created in Group Manager GUI (Optional)

Volume is online



Access is by the host iSCSI initiator name



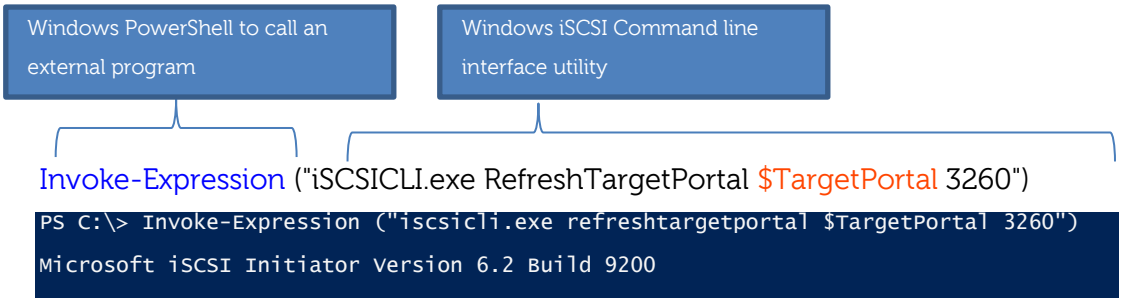
Note: Windows Server 2012 offers native iSCSI PowerShell cmdlets. See the Windows Server 2012 section for more details.

8. Now obtain the iSCSI target to the new volume using PS Series cmdlets. A cmdlet execution status indicator may pop up during processing.

```
#Get all the properties for this volume
PS C:\> $MyVol = Get-EqlVolume -VolumeName $ThisVol

#We only need the iSCSI Target name
PS C:\> $iTARGET=$MyVol.ISCSITargetName
```

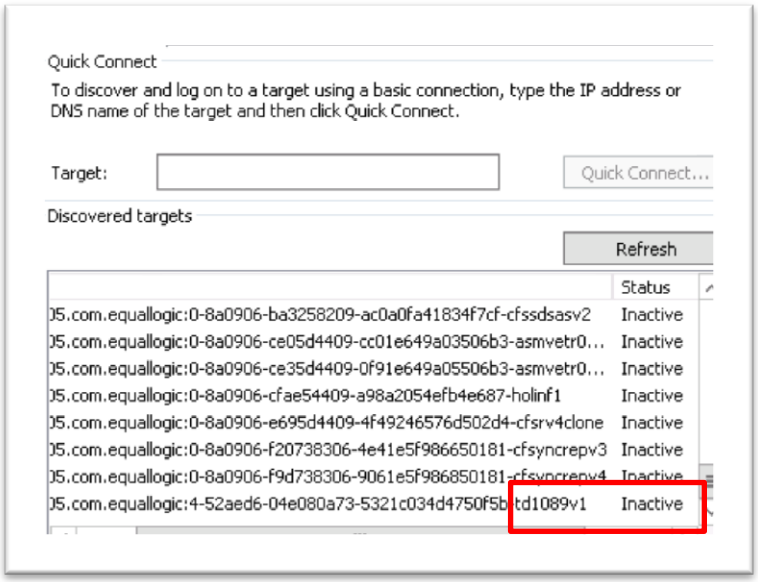
Note: the iSCSI Target Name will be used to login to our targets later in this exercise

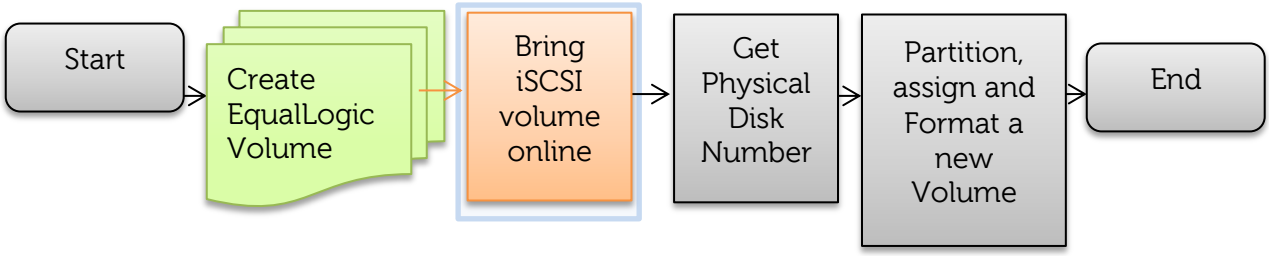
9. Refresh the portal with:



Note: The Target Portal can be obtained from the iSCSI Initiator tool under the "Discovery" tab. This will be the port used to discover all the iSCSI Targets (volumes).

The iSCSI Target should now show up as "inactive" in the iSCSI Initiator GUI.





10. You may also see the properties of the iSCSI session for just the volume created with this cmdlet:

Windows WMI utility

```
PS C:\ > Get-WMiObject  -Namespace root\wmi -Class
MSiSCSIInitiator_TargetClass|where-object {$_.TargetName -eq $iTARGET}
```

The volume is also known as the "target" in iSCSI terminology. $iTARGET was assigned earlier.

11. Next we will login the PS Series iSCSI sessions using the WMI interface
This command will retrieve all sessions for the volume we just created.

```
PS C:\> $objTarget = Get-WMiObject -computername $hostn -namespace
root\wmi -class MSiSCSIInitiator_TargetClass | where { $_.TargetName -eq
$iTARGET }

#Initialize the login options
PS C:\> $objLoginOpts = $null
```

Login to the target - first is the normal login, second adds it as a persistent target so that volume will persist between server reboots.

Windows WMI utility – login method

```
PS C:\>  $retn1 = $objTarget.Login($false, $null, $null, 0, $null,
$objLoginOpts, 0, $false)

PS C:\>  $retn2 = $objTarget.Login($false, $null, $null, 0, $null,
$objLoginOpts, 0, $true)
```

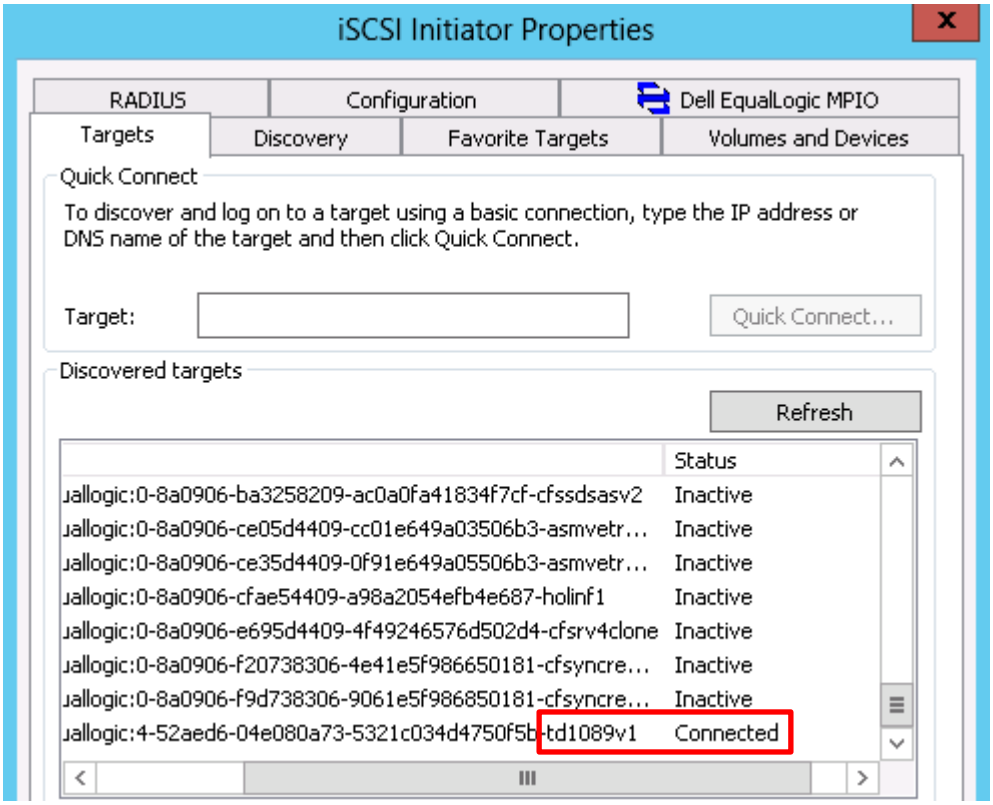
Quick Copy/Paste:

```
$objTarget = Get-WMiObject -computername $hostn -namespace root\wmi -class
MSiSCSIInitiator_TargetClass | where { $_.TargetName -eq $iTARGET }
$objLoginOpts = $null

$retn1 = $objTarget.Login($false, $null, $null, 0, $null, $objLoginOpts,
0, $false)
$retn2 = $objTarget.Login($false, $null, $null, 0, $null, $objLoginOpts,
0, $true)
```

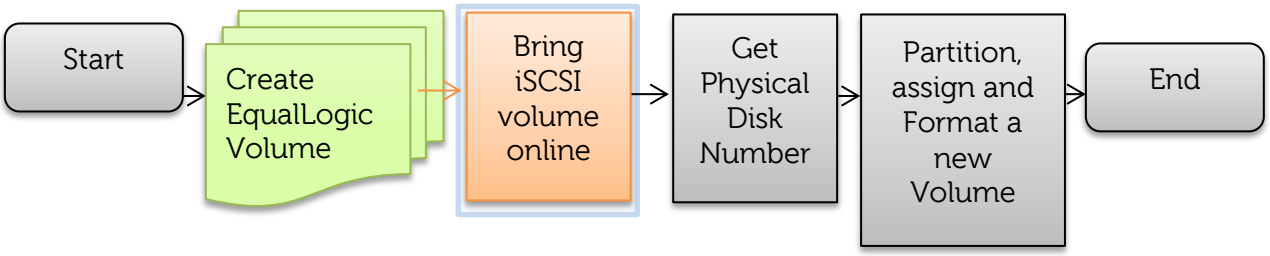
Paste the above into the command prompt. Hit enter. Example output below:

```
PS C:\> $objTarget = get-wmiobject -computername $hostn -namespace root\wmi -class
MSiSCSIInitiator_TargetClass | where { $_.TargetName -eq $iTARGET }
$objLoginOpts = $null
# Login to the target - first is the normal login, second adds it as a persistent
target so that volume will persist between server reboots
##
 $retn1 = $objTarget.Login($false, $null, $null, 0, $null, $objLoginOpts, 0,
$false)
 $retn2 = $objTarget.Login($false, $null, $null, 0, $null, $objLoginOpts, 0, $true)
```

12. The iSCSI Initiator GUI should now show connected:



Note: if the volume is still inactive try clicking on "Refresh"



13. Next, bring the volume online:

    Execute the WMI command to grab the iSCSI Sessions which will obtain the device information.

    The intent is to map the iSCSI session to the physical disk (WMI)

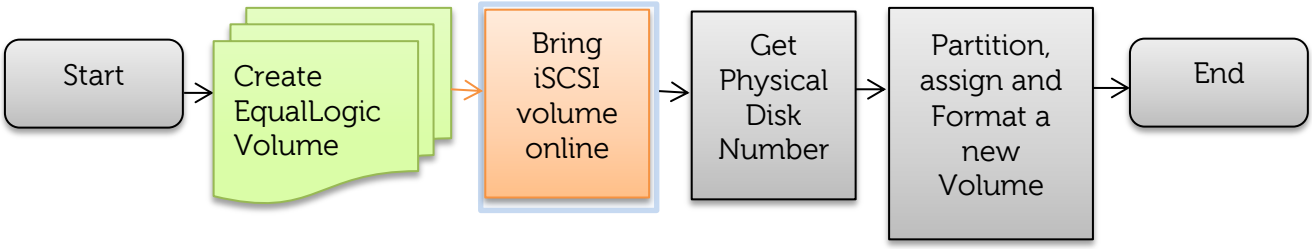
```
PS C:\> $colSessions = Get-WMIObject -namespace root\wmi -class
MSiSCSIInitiator_SessionClass|where-object {$_.TargetName -eq $iTARGET}
```

    Note: You may observe the output of the WMI command by removing the variable assignment at the beginning. We are interested in the ***Devices*** property (collection).

```
PS C:\> Get-WMIiobject -namespace root\wmi -class
MSiSCSIInitiator_SessionClass|where-object {$_.TargetName -eq $iTARGET}


__GENUS                : 2
__CLASS                : MSiSCSIInitiator_SessionClass
__SUPERCLASS           :
__DYNASTY              : MSiSCSIInitiator_SessionClass
__RELPATH              : MSiSCSIInitiator_SessionClass.SessionId="fffffa8004496020-
4000013700000004"
__PROPERTY_COUNT       : 8
__DERIVATION           : {}
__SERVER               : CFW2K12CC2
__NAMESPACE            : root\wmi
__PATH                 :
\\CFW2K12CC2\root\wmi:MSiSCSIInitiator_SessionClass.SessionId="fffffa8004496020-
4000013700000004"
ConnectionInformation : {1 0}
Devices                : {\\?\mpio#disk&ven_eqlogic&prod_100e-
00&rev_6.0_#1&7f6ac24&0&3634454432413335413738304530343542463735443433344330323135
3#{53f56307-b6bf-11d0-94f
                         2-00a0c91efb8b}}
InitiatorName          : iqn.1991-05.com.microsoft:cfw2k12cc2.spartan.local
ISID                   : {64, 0, 1, 55...}
SessionId              : fffffa8004496020-4000013700000004
TargetName             : iqn.2001-05.com.equallogic:4-52aed6-04e080a73-
5321c034d4750f5b-td1089v1
TargetNodeName         :
TSID                   : {160, 30}
PSComputerName         : CFW2K12CC2

__GENUS                : 2
__CLASS                : MSiSCSIInitiator_SessionClass
__SUPERCLASS           :
__DYNASTY              : MSiSCSIInitiator_SessionClass
__RELPATH              : MSiSCSIInitiator_SessionClass.SessionId="fffffa8004496020-
4000013700000005"
__PROPERTY_COUNT       : 8
__DERIVATION           : {}
__SERVER               : CFW2K12CC2
__NAMESPACE            : root\wmi
__PATH                 :
\\CFW2K12CC2\root\wmi:MSiSCSIInitiator_SessionClass.SessionId="fffffa8004496020-
4000013700000005"
ConnectionInformation : {1 0}
Devices                : {\\?\mpio#disk&ven_eqlogic&prod_100e-
00&rev_6.0_#1&7f6ac24&0&3634454432413335413738304530343542463735443433344330323135
3#{53f56307-b6bf-11d0-94f
```

```
                      2-00a0c91efb8b}}
InitiatorName         : iqn.1991-05.com.microsoft:cfw2k12cc2.spartan.local
ISID                  : {64, 0, 1, 55...}
SessionId             : fffffa8004496020-4000013700000005
TargetName            : iqn.2001-05.com.equallogic:4-52aed6-04e080a73-
5321c034d4750f5b-td1089v1
TargetNodeName        :
TSID                  : {4, 31}
PSComputerName        : CFW2K12CC2
```

Note: multiple sessions are due to the MPIO connections over NIC ports.



14. Since WMI iSCSI class is a collection of multiple sessions we will need to interrogate just the first instance to find the Device ID.

```
PS C:\> $objSession=$colSessions[0] #First instance of the Collection
```

Note: If only 1 session then remove the element index as so:
`$objSession=$colSessions`

15. Since the Devices property is embedded in these sessions we will have a collection as well. We only need the first instance to find the Device Number.

```
#Assign devices to a collection
PS C:\> $colDevices = $objSession.Devices
#Now we can reference the individual properties of the Devices
PS C:\> $objDevice=$colDevices[0] #Only need the first instance
```

To See the output simply reference the index of the $colDevices[0]. Good practice is to check all variables for values as you proceed.

```
PS C:\> $colDevices[0]


__GENUS             : 2
__CLASS             : MSiSCSIInitiator_DeviceOnSession
__SUPERCLASS        :
__DYNASTY           : MSiSCSIInitiator_DeviceOnSession
__RELPATH           :
__PROPERTY_COUNT    : 12
__DERIVATION        : {}
__SERVER            :
__NAMESPACE         :
__PATH              :
DeviceInterfaceGuid : 53f56307-b6bf-11d0-94f20a0c91efb8b
DeviceInterfaceName : \\?\mpio#disk&ven_eqlogic&prod_100e-
00&rev_6.0_#1&7f6ac24&0&3634454432413335413738304530343542463735443433344330323135
3#{53f56307-b6bf-11d0-94f2-0
                      0a0c91efb8b}
DeviceNumber        : 1
DeviceType          : 7
InitiatorName       : ROOT\ISCSIPRT\0000_0
LegacyName          : \\.\PhysicalDrive1
PartitionNumber     : 0
ScsiLun             : 0
ScsiPathId          : 0
ScsiPortNumber      : 3
ScsiTargetId        : 1
TargetName          : iqn.2001-05.com.equallogic:4-52aed6-04e080a73-
5321c034d4750f5b-td1089v1
PSComputerName      :
```

Note: If only one device exists in the collection remove the element index i.e. [0]. `$objDevice=$colDevices`

Quick Copy/Paste with logic (Steps 14-15):

Tip: Use "Shift+Enter" to edit each row in the command prompt.

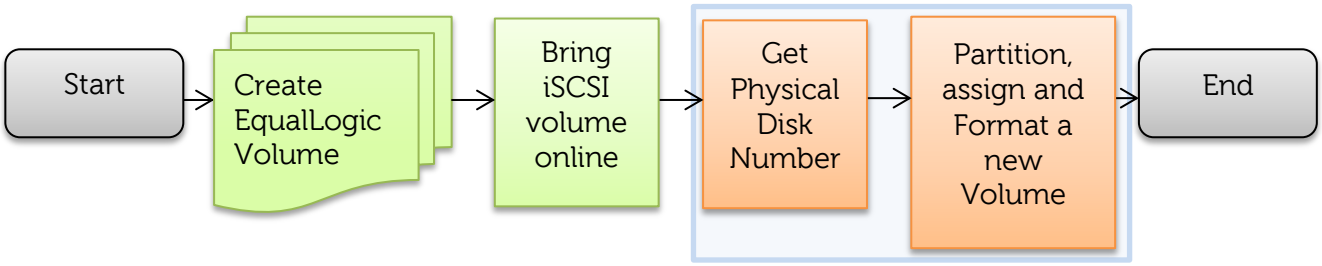
```
#Need only the single instance from the iSCSI connections

$objSession=$colSessions[0]
If (!$objSession -eq $Null){
Write-Host "Multiple MPIO sessions" $objSession

#OK with the single element

}
Else{
$objSession=$colSessions
Write-Host "Single MPIO session" $objSession
}
$colDevices=$colDevices = $objSession.Devices


#Now for the single Device element

$objDevice=$colDevices[0]

#Again Check for single or multiple sessions

If (!$objDevice -eq $Null){
Write-Host "Multiple MPIO sessions and device entries" $objDevice

#OK with the single element

}
Else{
$objDevice=$colDevices
Write-Host "Single MPIO session and this is the PhysicalDevice#:"
$objDevice.DeviceNumber
}
```

16. Build a meaningful label matching the EQL Volume/Pool and Group

```
#Example of substring parsing

PS C:\ > $label = $objSession.targetname.Substring(63) + $PoolName +
$GroupName
```

Or:

```
#Little easier to use the volume properties

PS C:\> $label = $MyVol.VolumeName + $PoolName + $GroupLabel
```



17. Now reference the "Device Number" and build the DISKPART script

    Copy/Paste block:

```
$diskID = $objDevice.DeviceNumber #Get the physicaldisk
$dpscript = @"
        select disk $diskID
        online disk noerr
        ATTRIBUTES DISK CLEAR READONLY
        clean
        convert mbr
        create partition primary
        select part 1
        format fs=NTFS label=`"$label`" quick nowait noerr`n
        assign letter=E
"@
$dpscript | diskpart #Executes the above commands
```

    Note: Be sure to use a volume letter that is not in use

    Copy/Paste into the command line and hit enter as demonstrated below:

```
PS C:\> $dpscript = @"
        select disk $diskID
        online disk noerr
        ATTRIBUTES DISK CLEAR READONLY
        clean
        convert mbr
        create partition primary
        select part 1
        format fs=NTFS label=`"$label`" quick nowait noerr`n
        assign letter=E
"@
$dpscript | diskpart #Executes the above commands
```

    The output will show each DISKPART command execution status:

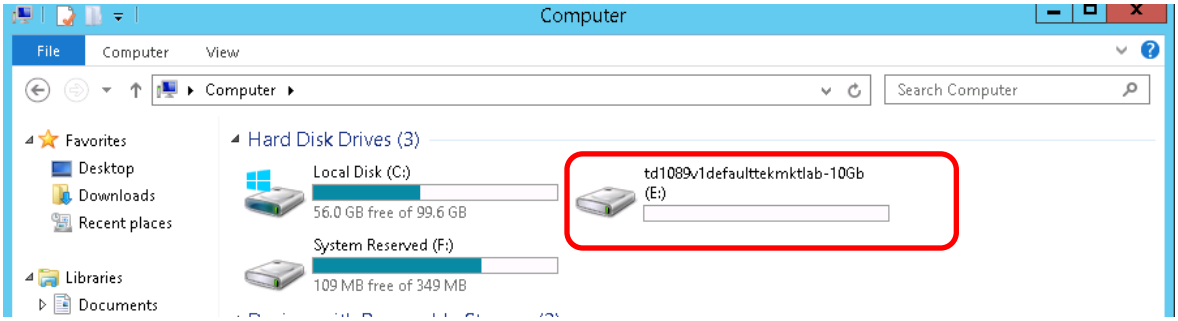
```
Now initializing and formatting (bringing online) PS Array volume  TD1089V1  to
Windows volume  td1089v1defaulttekmktlab-10Gb  Disk#: 1

Microsoft DiskPart version 6.2.9200

Copyright (C) 1999-2012 Microsoft Corporation.
On computer: CFW2K12CC2

DISKPART>
Disk 1 is now the selected disk.

DISKPART>
DiskPart successfully onlined the selected disk.

DISKPART>
Disk attributes cleared successfully.

DISKPART>
DiskPart succeeded in cleaning the disk.

DISKPART>
DiskPart successfully converted the selected disk to MBR format.
```

```
DISKPART>
DiskPart succeeded in creating the specified partition.

DISKPART>
Partition 1 is now the selected partition.

DISKPART>


The format has been initiated successfully and is currently in progress.
Please wait until format has completed before trying to access the volume.

DISKPART>
DISKPART>
DiskPart successfully assigned the drive letter or mount point.

DISKPART>
```

Note: Occasionally a Format request may prompt. Click "Cancel" since the volume is already formatted.

Windows Server 2012 offers native Storage PowerShell cmdlets. See the Windows Server 2012 section for more details.

18. Completed the automation of creating a volume on a PS Series Array and the volume is now available for use.



For the complete PowerShell script please click here copy/paste and make the appropriate changes to match the PS Series configuration.

Windows PowerShell Dell EqualLogic PS Series Volume Deletion and iSCSI Connection removal

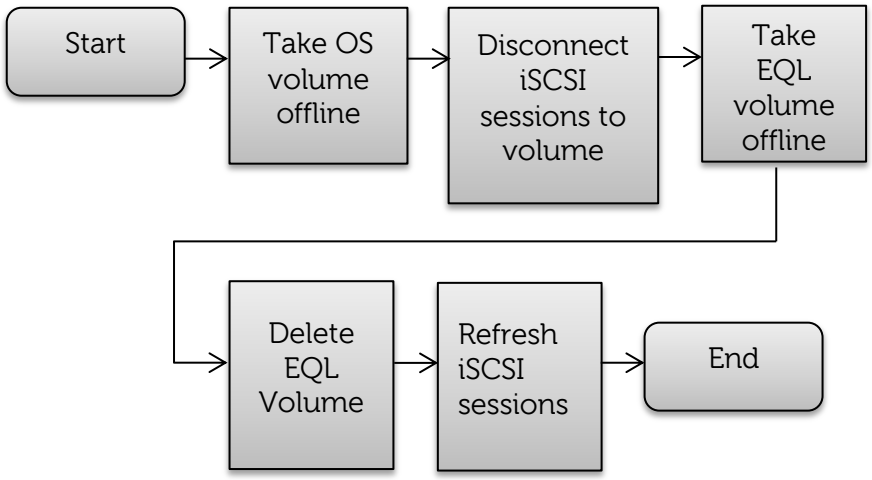
In this section we will show how to take the volume offline, disconnect the iSCSI sessions and remove the volume from the PS Series Array. By providing both the volume create and volume delete process automation testing will be a simpler task. This is the method required for Windows operating systems prior to Windows Server 2012/Windows 8. Windows Server 2012/Windows 8 have additional integration and capabilities with Dell EqualLogic storage.

However the below scripts will work with Windows Server 2012, Windows 8 and Windows 2008.

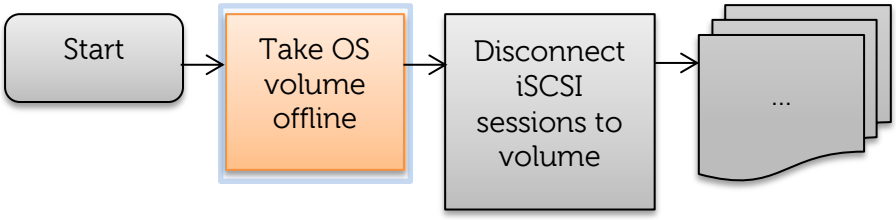
The example that follows will use the Dell™ PS Series PowerShell cmdlets, Windows PowerShell cmdlets, Windows iSCSI CLI (iSCSI session management utility), Windows Management Interface (WMI) as well as DISKPART utility for Windows.

To delete a volume through a command line interface the following steps will be executed.

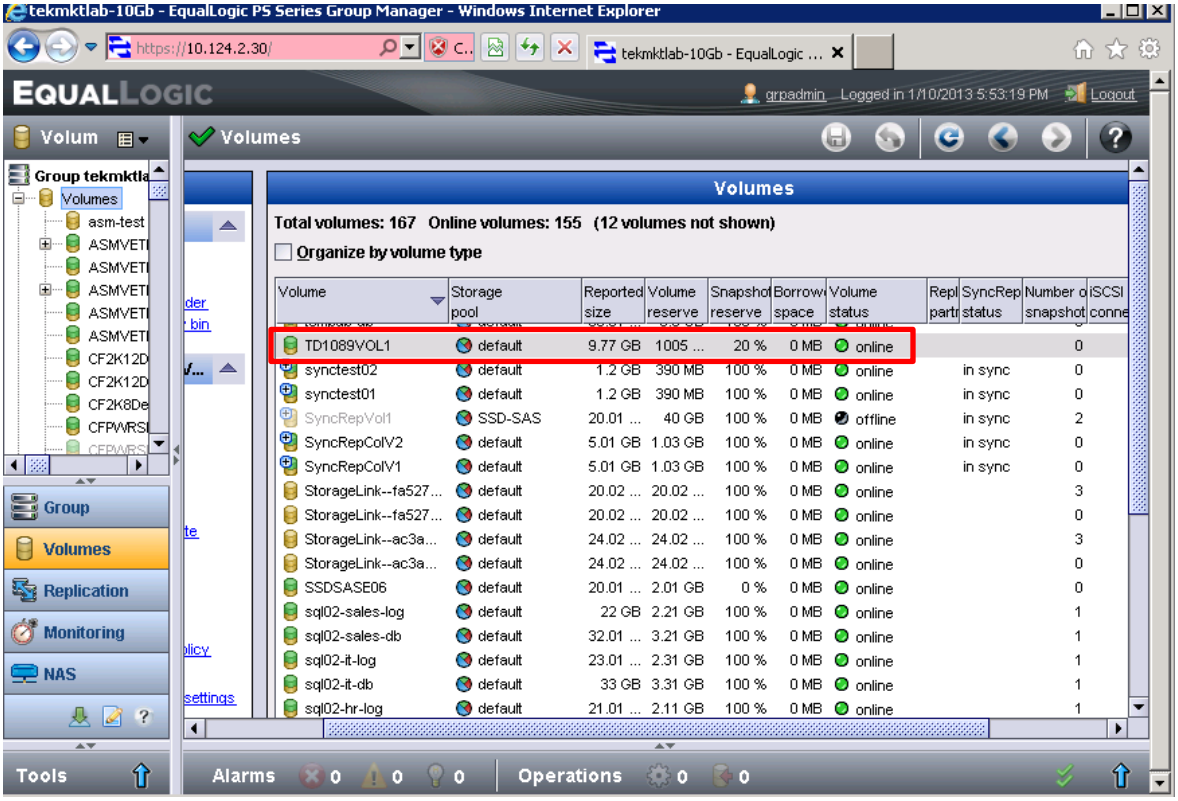
1. Use the Windows DISKPART utility to take the Windows volume "Offline"
2. Use WMI commands to remove all iSCSI Sessions to the volume
3. Use PS Series PowerShell cmdlets to take the PS Series volume "Offline"
4. Use PS Series PowerShell cmdlets to remove the PS Series volume
5. Refresh the iSCSI Targets with the Windows "iSCSIcli.exe" utility



Step – By – Step Pre-Windows Server 2012



1. For this exercise check the current status of the volume in the PS Series Group Administrator GUI. Launch Group Manager and locate the volume you wish to remove.



Note: This step is intended to familiarize the reader with the scripts and may be skipped during actual automation executions.

2. For simplicity, set up the variables.

```
PS C:\> $ThisVol = 'MyVol' #change to volume you created earlier
$TargetPortal= '127.0.0.1' #change to the group ip
```

Note: This step may be skipped if these variables are already set from executing the steps to create a volume.

3. Retrieve the iSCSI Target to allow the setting of the Windows volume to offline.

Use PS Series cmdlet to get the volume

```
PS C:\> $MyVol = Get-EqlVolume -VolumeName $ThisVol
```

Associate the iSCSI target to pass to the WMI command.

```
PS C:\> $iTARGET=$MyVol.ISCSITargetName
```

Retrieve all MPIO sessions to this volume with WMI.

```
PS C:\> $colSessions = get-wmiobject -namespace root\wmi -class
MSiSCSIInitiator_SessionClass|where-object {$_.TargetName -eq $iTARGET}
```

The MPIO sessions may be listed with the Get-WMiObject command as demonstrated below.

```
PS C:\ > Get-WMiObject -namespace root\wmi -class
MSiSCSIInitiator_SessionClass|where-object {$_.TargetName -eq $iTARGET}


__GENUS                 : 2
__CLASS                 : MSiSCSIInitiator_SessionClass
__SUPERCLASS            :
__DYNASTY               : MSiSCSIInitiator_SessionClass
__RELPATH               : MSiSCSIInitiator_SessionClass.SessionId="fffffa8004496020-
4000013700000060"
__PROPERTY_COUNT        : 8
__DERIVATION            : {}
__SERVER                : CFW2K12CC2
__NAMESPACE             : root\wmi
__PATH                  :
\\CFW2K12CC2\root\wmi:MSiSCSIInitiator_SessionClass.SessionId="fffffa8004496020-
4000013700000060"
ConnectionInformation : {1 0}
Devices                 : {\\?\mpio#disk&ven_eqlogic&prod_100e-
00&rev_6.0_#1&7f6ac24&0&363039304130393830424138423736393846323532354535424142383431#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}}
InitiatorName           : iqn.1991-05.com.microsoft:cfw2k12cc2.spartan.local
ISID                    : {64, 0, 1, 55...}
SessionId               : fffffa8004496020-4000013700000060
TargetName              : iqn.2001-05.com.equallogic:0-8a0906-768bba009-
41b8bae525250f98-td1089v1
TargetNodeName          :
TSID                    : {35, 40}
PSComputerName          : CFW2K12CC2

__GENUS                 : 2
__CLASS                 : MSiSCSIInitiator_SessionClass
__SUPERCLASS            :
__DYNASTY               : MSiSCSIInitiator_SessionClass
__RELPATH               : MSiSCSIInitiator_SessionClass.SessionId="fffffa8004496020-
4000013700000063"
__PROPERTY_COUNT        : 8
__DERIVATION            : {}
__SERVER                : CFW2K12CC2
__NAMESPACE             : root\wmi
__PATH                  :
\\CFW2K12CC2\root\wmi:MSiSCSIInitiator_SessionClass.SessionId="fffffa8004496020-
4000013700000063"
ConnectionInformation : {1 0}
Devices                 : {\\?\mpio#disk&ven_eqlogic&prod_100e-
00&rev_6.0_#1&7f6ac24&0&363039304130393830424138423736393846323532354535424142383431#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}}
InitiatorName           : iqn.1991-05.com.microsoft:cfw2k12cc2.spartan.local
ISID                    : {64, 0, 1, 55...}
SessionId               : fffffa8004496020-4000013700000063
TargetName              : iqn.2001-05.com.equallogic:0-8a0906-768bba009-
41b8bae525250f98-td1089v1
TargetNodeName          :
TSID                    : {37, 40}
PSComputerName          : CFW2K12CC2
```

This example shows two MPIO sessions.

4. Only need one session to take the Windows volume offline with DISKPART.

```
#Need only the single instance from the iSCSI connections
PS C:\ > $objSession=$colSessions[0]
#For only one nic or session
PS C:\ > $objSession=$colSessions
PS C:\ > $colDevices = $objSession.Devices
#Need to interrogate only the first Devices collection
PS C:\ > $objDevice=$colDevices[0]
#For only one nic or session
PS C:\ > $objDevice=$colDevices
```

Quick Copy/Paste with logic:

Tip: Use "Shift+Enter" to edit each row in the command prompt.

```
#Need only the single instance from the iSCSI connections

$objSession=$colSessions[0]
If (!$objSession -eq $Null){
Write-Host "Multiple MPIO sessions" $objSession

#OK with the single element

}
Else{
$objSession=$colSessions
Write-Host "Single MPIO session" $objSession
}
$colDevices=$colDevices = $objSession.Devices


#Now for the single Device element

$objDevice=$colDevices[0]

#Again Check for single or multiple sessions

If (!$objDevice -eq $Null){
Write-Host "Multiple MPIO sessions and device entries" $objDevice

#OK with the single element

}
Else{
$objDevice=$colDevices
Write-Host "Single MPIO session and this is the PhysicalDevice#:"
$objDevice.DeviceNumber
}
```

5. Setup for the DISKPART utility. This will identify the device number that is online to this volume:

```
PS C:\> $diskID = $objDevice.DeviceNumber
PS C:\> Write-Host "Will vary " $ThisVol " Disk#:" $disknum " Offline"
```

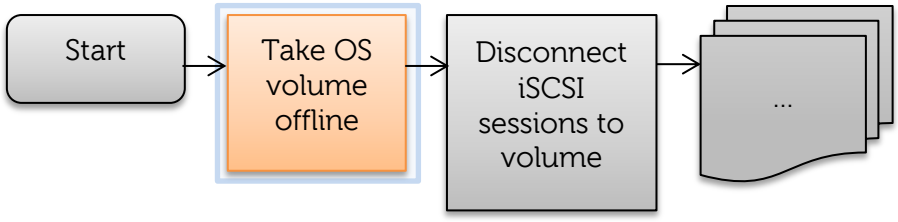
Next set the variable for the commands to pipe to DISKPART:

Copy/Paste

```
$dpscript = @"
    select disk $diskID
    clean
    offline disk noerr
"@
```

```
Select the above and paste into your PowerShell command line and <enter>:
```

```
PS C:\ > $dpscript = @"
    select disk $diskID
    clean
    offline disk noerr
"@
```

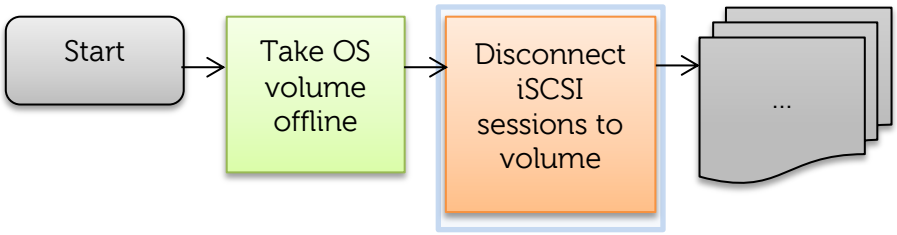


And now to execute DISKPART:

```
PS C:\> $dpscript | diskpart
```

Output will look similar to below:

```
PS C:\ > $dpscript | diskpart

Microsoft DiskPart version 6.2.9200

Copyright (C) 1999-2012 Microsoft Corporation.
On computer: CFW2K12CC2

DISKPART>
Disk 1 is now the selected disk.

DISKPART>
DiskPart succeeded in cleaning the disk.

DISKPART>
DiskPart successfully offlined the selected disk.
```



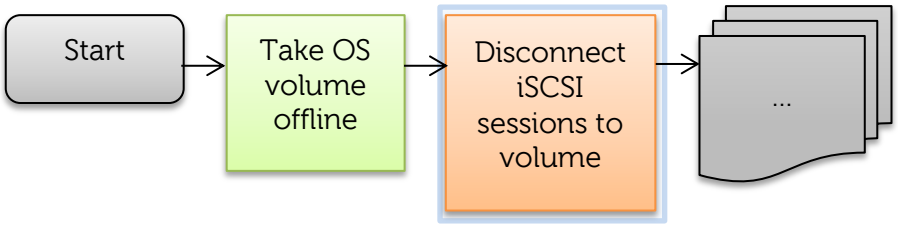
6. Next logout of each iSCSI Session connected to the volume. The iSCSI targets are still associated to the *$colSessions* variable assigned earlier with the WMI command (see step 3):

```
PS C:\> foreach ($objSession in $colSessions)
    {
        write-host "Host $hostn - logging out ISCSI target $iTARGET"
        $retn = $objSession.Logout()
    }
```

The entire "ForEach" statement may be pasted and then <enter> into the command interface as demonstrated. Output will look similar to below:

```
PS C:\ > foreach ($objSession in $colSessions)
    {
        write-host "Host $hostn - logging out ISCSI target $iTARGET"
        $retn = $objSession.Logout()
    }
Host CFW2K12CC2 - logging out ISCSI target iqn.2001-05.com.equallogic:0-8a0906-
097bba009-7528bae58ce50f9a-td1089v1
Host CFW2K12CC2 - logging out ISCSI target iqn.2001-05.com.equallogic:0-8a0906-
097bba009-7528bae58ce50f9a-td1089v1

PS C:\ >
```

Note: ForEach is used by PowerShell to iterate through a collection of objects.



7. Remove the persistent connections – this time the persistent login class will be used "MSiSCSIInitiator_PersistentLoginClass" for the WMI command:

```
PS C:\> $colPersistent = get-wmiobject -computername $hostn -namespace
root\WMI -class MSiSCSIInitiator_PersistentLoginClass | where {
$_.TargetName -eq $iTARGET }
```

Remove the variable to see the assignment. Output will look similar to below:

```
Get-WMIobject -computername $hostn -namespace root\WMI -class
MSiSCSIInitiator_PersistentLoginClass | where { $_.TargetName -eq $iTARGET
}
```

```
PS C:\> get-wmiobject -computername $hostn -namespace root\WMI -class
MSiSCSIInitiator_PersistentLoginClass | where { $_.TargetName -eq $iTARGET }


__GENUS                 : 2
__CLASS                 : MSiSCSIInitiator_PersistentLoginClass
__SUPERCLASS            :
__DYNASTY               : MSiSCSIInitiator_PersistentLoginClass
__RELPATH               :
MSiSCSIInitiator_PersistentLoginClass.TargetName="iqn.2001-05.com.equallogic:0-
8a0906-097bba009-7528bae58ce50f9a-td1089v1"
__PROPERTY_COUNT        : 8
__DERIVATION            : {}
__SERVER                : CFW2K12CC2
__NAMESPACE             : root\WMI
__PATH                  :
\\CFW2K12CC2\root\WMI:MSiSCSIInitiator_PersistentLoginClass.TargetName="iqn.2001-
05.com.equallogic:0-8a0906-097bba009-7528bae58ce50f9a-td1089v1"
InitiatorInstance       : ROOT\ISCSIPRT\0000_0
InitiatorPortNumber     : 4294967295
IsInformationalSession  : False
LoginOptions            : System.Management.ManagementBaseObject
Mappings                :
SecurityFlags           : 0
TargetName              : iqn.2001-05.com.equallogic:0-8a0906-097bba009-
7528bae58ce50f9a-td1089v1
TargetPortal            : System.Management.ManagementBaseObject
PSComputerName          : CFW2K12CC2
```

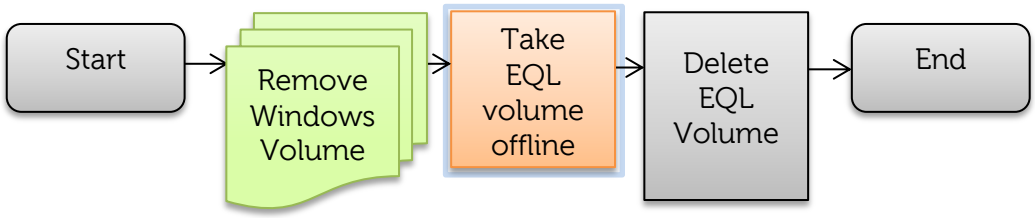
8. Iterate through all sessions and logout and delete each iSCSI session to the target.

```
PS C:\ > foreach ($objPersistent in $colPersistent)
    {
write-host "Host $hostn - removing persistent target:"
$objPersistent.TargetName
$objPersistent.psbase.Delete()
}
```

Note: $colPersistent was assigned in step 7 using the WMI "Persistent Login Class".

Output will look similar to below:

```
PS C:\ > foreach ($objPersistent in $colPersistent)
    {
        write-host "Host $hostn - removing persistent target:"
$objPersistent.TargetName
        $objPersistent.psbase.Delete()
    }
Host CFW2K12CC2 - removing persistent target: iqn.2001-05.com.equallogic:0-8a0906-
097bba009-7528bae58ce50f9a-td1089v1

PS C:\ >
```



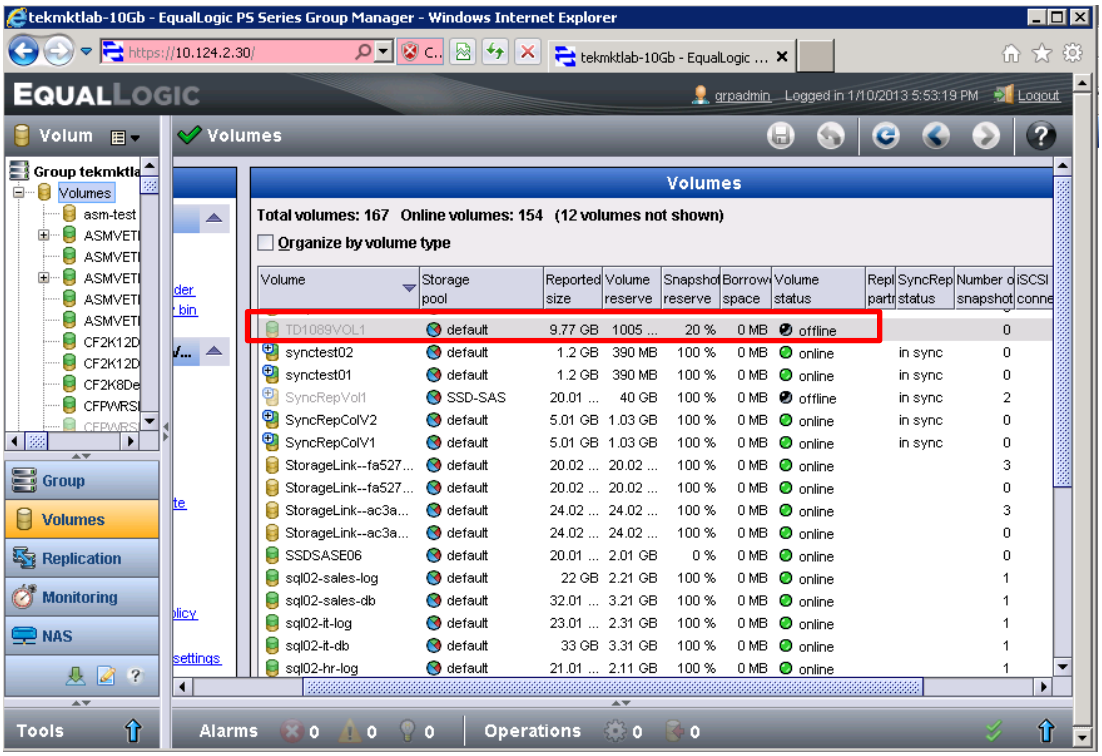
9. Next the PS Series Array volume will need to be brought offline then removed.

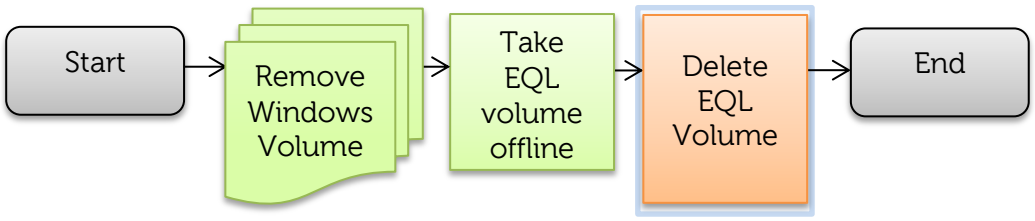
```
PS C:\> set-eqlvolume -volumename $ThisVol -OnlineStatus offline
```

Command output in the Windows PowerShell ISE

```
PS C:\ > set-eqlvolume -volumename $ThisVol -OnlineStatus offline
Volume 'TD1089VOL1' changed successfully
```

Optionally, use EqualLogic Group Administrator GUI to check the volume status.





10. Now remove the volume

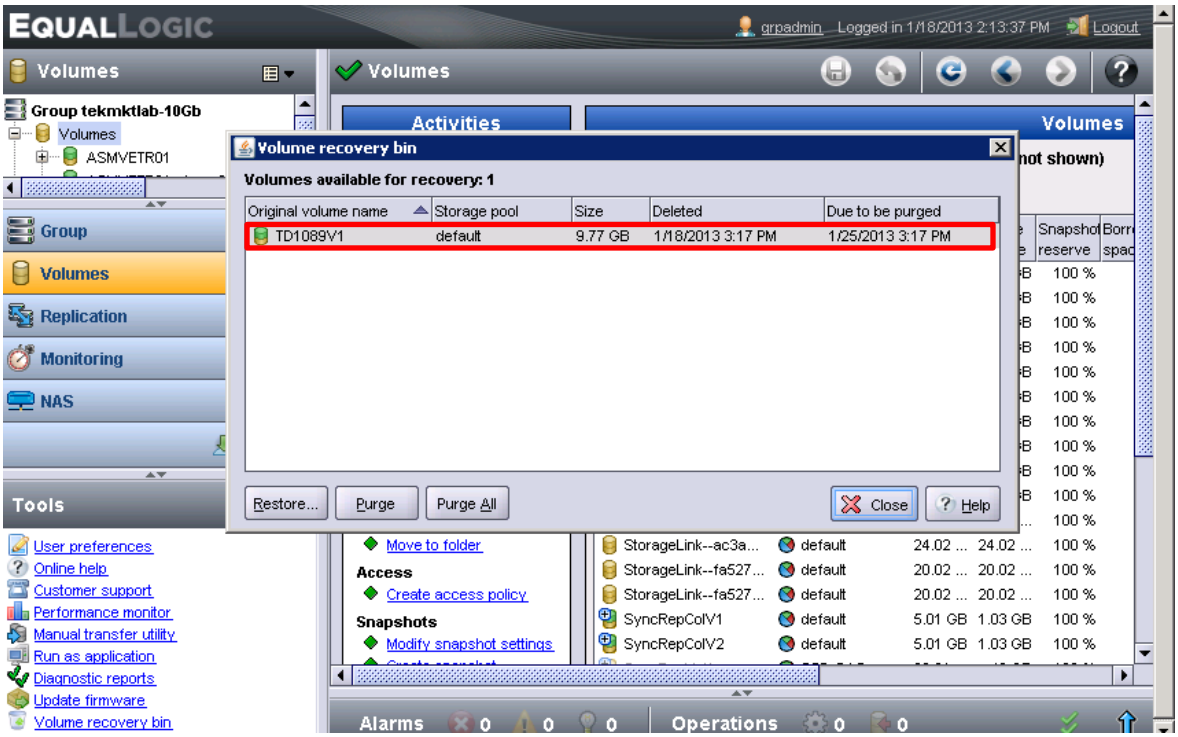
```
PS C:\> Remove-EqlVolume -VolumeName $ThisVol -Force
```

Note: "-Force" will bypass the confirmation message

Output will look similar to below:

```
PS C:\> Remove-EqlVolume -VolumeName $ThisVol -Force
PSAPI.Cmdlets.RemoveVolume TD1089VOL1 completed successfully.
```

Notice the deleted volume in the PS Series Group Manager GUI "Volume recovery bin" available on PS Series firmware v6 and up which provides the ability to restore the volume if necessary.



For the complete PowerShell script please click here copy/paste and make the appropriate changes to match the environment and PS Series configuration.

Windows Server 2012 Native Windows Storage Management Automation Techniques

Windows Server 2012 and Windows 8 introduced a new Storage Management Architecture to help reduce management complexity as well as optimize storage through native Windows Storage Management Applications.

Host Integration Tools for Microsoft v4.5 and higher (HIT/Microsoft) includes a new Storage Management Provider (SMP) for Windows 8 and Windows Server 2012. This is the preferred framework for enabling support for Windows-based storage management. This new framework utilizes the Windows Storage Management APIs (SM API) which allows for discovery, provisioning and protection of Dell PS Series storage.

The Dell EqualLogic SMP allows for management of Dell EqualLogic storage directly through native Windows storage interfaces such as storage PowerShell cmdlets (Storage Module), the File Services user interface (UI) in the Windows 2012 Server Manager console, the standard Windows Management Instrumentation API as well as Systems Center Virtual Machine Manager 2012 SP1 SCVMM (with the HIT for Microsoft v4.6).

The Storage Management Application Interface (SM API) and SMP infrastructures are native to Windows 8 and Windows Server 2012.

The Dell EqualLogic SMP is an optional component installed by default by the HIT/Microsoft installer. The provider is hosted by the Dell EqualLogic SMP Host Service (EqlSMPHost).

Add volumes example with native Windows Server 2012 cmdlets

The following example requires Windows Server 2012 or Windows 8 and the Dell EqualLogic Host Integration Tools for Microsoft v4.5 or higher.

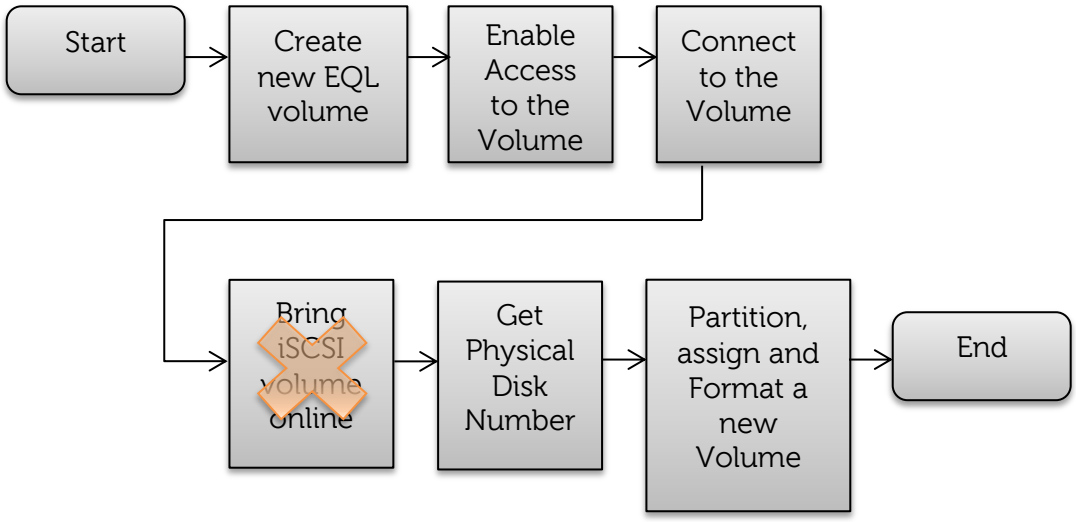
To add volumes to a PS Series group using native Windows Server 2012 cmdlets the following steps will be executed.

- Create new volume(s) on the PS Series Array through the native Windows Server 2012 PowerShell cmdlets.
- Enable the masking set for that volume to the host initiator (to provide access to the volume through the iSCSI network)
- Connect the volume(s) created on the PS Series array to the host with Windows Server 2012 PowerShell cmdlets
- Bring the new volume online and format a new NTFS volume assigned to the next available drive letter.
- Introduce some methods of repetitive processing with PowerShell

The steps to create a volume with Native Window Server 2012 are similar to Windows 2008 with EqualLogic cmdlets only the implementation is different because of the native Windows integration with Dell EqualLogic storage as well as the maturity of PowerShell with iSCSI and Storage related cmdlets.

Add volumes example with native Windows Server 2012 cmdlets

The details we needed in Windows 2008 to just retrieve the Physical disk number to initialize, partition and format are much easier to obtain with the new Storage Architecture of Windows Server 2012 Also the new iSCSI Session cmdlets connect and bring the volume online consolidating some of the steps.



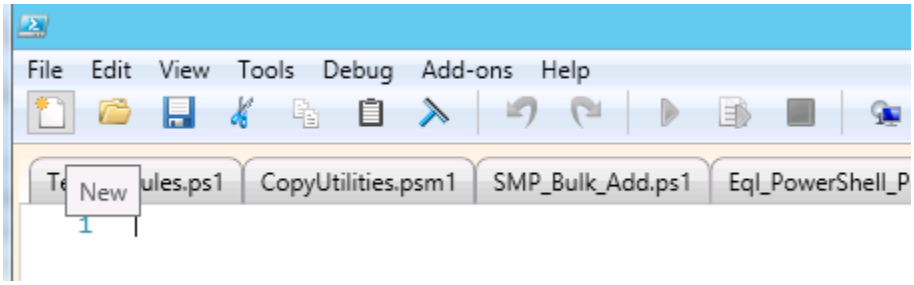
Launch the Windows ISE and create a new PowerShell script (File -> New) and copy and paste the automated volume creation script.

Note: Please see the section on PS Series Group Access and the section to delete the volume if needed. The PS Series group must be connected with appropriate access (please see the Dell EqualLogic PS Series Group Access section).

Warning:

Below is for Reference only and should only be used with careful consideration of the impact. Not all exceptions may be captured.

In Windows Server 2012 PowerShell ISE create a new page



IMPORTANT: This PowerShell script as shown will create 3 volumes, connect and bring them online. Automatically creating volumes may have an impact so the recommendation is to set the *$totvols* initially to 1 so only 1 volume is created (note the corresponding remove volume script may need to be changed as well).

Add volumes example with native Windows Server 2012 cmdlets

Copy/Paste the code below with the appropriate changes into the new page:

```
#-------------------------------------------------------------------------
#  Copyright (c) 2011-2013 by Dell Inc.
#
# All rights reserved.  This software may not be copied, disclosed,
# transferred, or used except in accordance with a license granted
# by Dell Inc.  This software embodies proprietary information
# and trade secrets of Dell Inc.
#
#-------------------------------------------------------------------------

#volume add using all native windows cmdlets
#Please change below to match your environment

# This script uses a simple for statement to create the number of volumes
# automatically...many other methods could be employed such as have a .csv file
# with all the parameters listed
# PLEASE USE WITH CAUTION

#the volume name pattern...a number will be appended
$myTVol="MyEQLvol"
#The number of volumes starting with 0
$totVols = 0
#This is the group that should have been initialized
# connect-eqlgroup or new-eqlgroupaccess should have already been performed
$myGroup ="MyGroupName"
#will create a thin volume with capacity specified
$useable = "10GB"
$myVol = ""
#Specify minimium free GB before allocating volumes
$MinFree = 1000
#Adjust the free to account for the new volumes 10 is the usable in GB for each
vol + 10%
[int]$MinFree=[int]$MinFree + ((10*([int]$totVols + 1)) * 1.1)

$MyPool = "MyPool"

#Utility functions
function Get-NextFreeDriveLetter
{
    return (Get-ChildItem function:[d-z]: -n | ?{ !(test-path $_) } | Get-
Random)
}

Function Div([Parameter(Mandatory=$true)][double]$x,
[Parameter(Mandatory=$true)][double]$y) { $x / $y }



#One time grabs for the initiator port
$initaddress=(get-initiatorport)
$tarport=(get-targetportal)

#The -ErrorAction SilentlyContinue would simply give you the ability to decide
what is written to the console
#See the condition checks

$GroupIsOK= get-storagesubsystem -FriendlyName $myGroup -ErrorAction
SilentlyContinue
$PoolIsOK = get-storagepool -FriendlyName $MyPool -ErrorAction SilentlyContinue

#Note: additional logic is needed if more than one pool exists with the same
name (i.e. more than one group
#is connected)
```

```
#CHECK ARRAY AND POOL FOR OPERATIONS AND/OR ACCESS VIA HIT/ME
if ($GroupIsOK.OperationalStatus -eq "OK")
 {
 write-host $GroupIsOK.FriendlyName $GroupIsOK.FirmwareVersion $GroupIsOK.Model

 }
 else
 {
 #The exit with rc could be interrogated by $LASTEXITCODE
 #this method of exit on validation fail is not necessarily a best coding
practice however easy to understand
 write-host "Install the EqualLogic Host Integration Tools for Microsoft and set
access and connect to the group."
 exit 33
  }


[int]$FreeSpace = $PoolIsOK[0].AllocatedSize / [int](1024*1024*1024) #GB

if ($PoolIsOK.OperationalStatus -eq "OK"){
    if ($FreeSpace -le $MinFree){
    write-host "Not Enough Free space in the pool " $PoolIsOK.FriendlyName
$FreeSpace.ToString() "GB" $PoolIsOK.HealthStatus
    exit 35
    }
#Enough Free space and ok
[int]$FreeSpace = $PoolIsOK.AllocatedSize/1024/1024/1024/1024
write-host $PoolIsOK.FriendlyName $FreeSpace.ToString() "TB"
$PoolIsOK.HealthStatus

}
else
{
Write-Host "Please be sure you have access to this pool"
exit 34
}

#Ok passed some of the initial tests now for creating the volumes
Write-Host "Now we will create " $totVols " volumes to " $mygroup " from "
$initaddress.NodeAddress


for ($i=0;$i -le $totVols;$i++){

$myVol = $myTVol + $i
Write-Host "Creating a volume " $myVol " in " $MyPool " on EqualLogic group "
$myGroup

New-VirtualDisk -FriendlyName $myVol -Size 10GB -ProvisioningType Thin -
StoragePoolFriendlyName  $MyPool

Write-Host "Now adding the masking set to allow for this volume to be accessed
by this local Server:" $initaddress.NodeAddress " to " $mygroup
New-MaskingSet -StorageSubSystemFriendlyName $myGroup -VirtualDiskNames $myVol -
InitiatorAddresses $initaddress.NodeAddress

#The Show-VirtualDisk cmdlet makes a virtual disk available to a host (by
initiator and target ports).
Show-VirtualDisk -FriendlyName $myVol -TargetPortAddresses $tarport.PortNumber -
InitiatorAddress $initaddress.NodeAddress
}


Write-Host "Now we will refresh the iscsi initiator which may take a few
minutes.."

#Below will take a considerable time for many targets....
##Update-IscsiTarget
#The alternative is to update the portal...much more efficient
#This will work if only one target portal is available...check iSCSI Initiators
GUI on the
```

Add volumes example with native Windows Server 2012 cmdlets

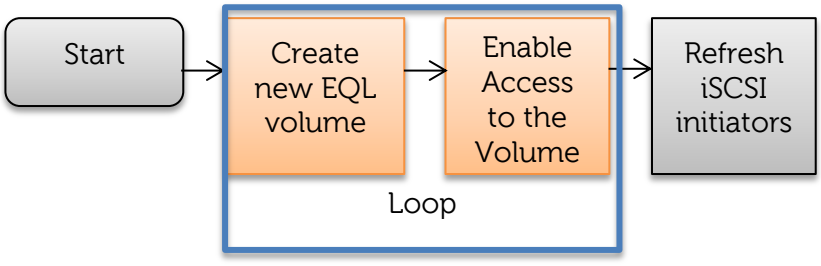
```
# "Discovery" tab
Get-iSCSITargetPortal | Update-iSCSITargetPortal

Write-Host "Now we will connect each new volume to this host:"
$initaddress.NodeAddress " to " $mygroup

$myVol=""


for ($i=0;$i -le $totVols;$i++){

$myVol = $myTVol + $i
Write-Host "Connecting" $initaddress.NodeAddress " to " $myVol
#Kinda fancy way to get the target node address
$IQN = (Get-VirtualDisk -FriendlyName $myVol | Get-TargetPort).NodeAddress

$iSCSISession = Connect-iSCSITarget -NodeAddress $IQN

$disk= ($iSCSIsession| Get-Disk) #pipes the isci target to get the physical disk

initialize-disk -InputObject $disk

#partition and assign the next available drive letter

$partition = New-Partition -InputObject $disk -UseMaximumSize

#format the volume for access
Format-Volume -Partition $partition -FileSystem NTFS -NewFileSystemLabel $myVol
-Confirm:$false

#partition will pop-up format so do that latter..
$Letter = (Get-NextFreeDriveLetter)

$partition | Add-PartitionAccessPath -Accesspath $Letter

}
#List the volumes
Get-Volume
Write-Host "Completed creating volumes on EqualLogic Storage using SMP
integration"
```

Save the PowerShell script as "EQL_W2K12_BULK_VOL_CREATE.ps1" or other name as appropriate and execute through the ISE or at the command line.

To execute through the Windows PowerShell ISE click the green arrow  or from the Menu click Debug-> Run Continue. You may set breakpoints at any execution line for further interrogation.

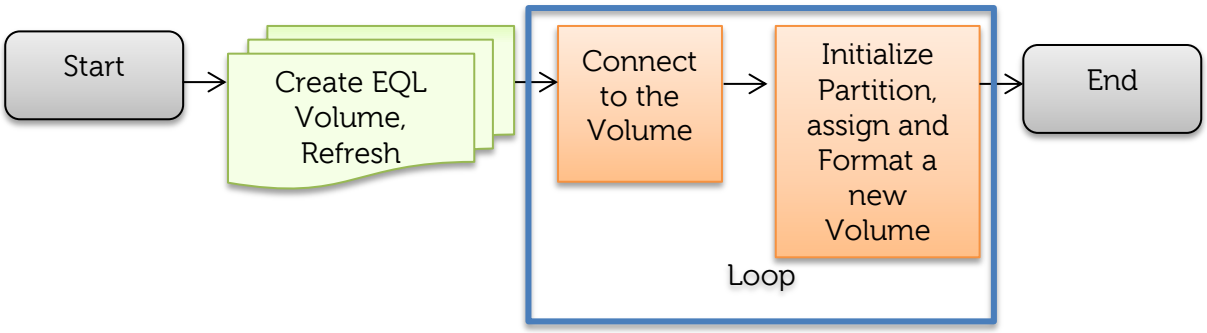
Explanation of the steps above:

Similar to the Windows 2008 example the array is interrogated for available free space in the pool for the new volumes we intend to create. Also, we can check for the communication to the group before the create volume loop which will use: *New-VirtualDisk* to create the EqualLogic volume, *New-MaskingSet* to allow for the host to access this volume from the PS Series perspective and then *Show-VirtualDisk* to indicate the access from the Windows host perspective.



Add volumes example with native Windows Server 2012 cmdlets

The second part of the script will simply loop through each volume created and connect the iSCSI sessions with the new iSCSI Session cmdlet *Connect-iSCSITarget*. Then the volumes will be initialized with *Initialize-Disk*, partitioned with *New-Partition Format-Volume* then assign the next drive letter with *Add-PartitionAccessPath*.



Example Partial Output below:

```
PS C:\> .\EQL_W2K12_BULK_VOL_CREATE.ps1
tekmktlab-10Gb Storage Array Firmware V6.0.2 (R287892) PS6010, PS6110
default 24 TB Healthy
Now we will create  2  volumes to  tekmktlab-10Gb  from  iqn.1991-05.com.microsoft:cfscvmm2k12sp1.spartan.lo
Creating a volume  CFSMPvol0  in  default  on EqualLogic group  tekmktlab-10Gb

FriendlyName            ResiliencySettingName   OperationalStatus      HealthStatus           IsManualAttach
Size
------------            ---------------------   -----------------      ------------           -------------
----
CFSMPvol0                                       OK                     Healthy
10 GB
Now adding the masking set to allow for this volume to be accessed to by Server: iqn.1991-
05.com.microsoft:cfscvmm2k12sp1.spartan.local  to  tekmktlab-10Gb

ObjectId            : {1}\\CFSCVMM2K12SP1\root/Microsoft/Windows/Storage/Providers\EQL_MaskingSet.ObjectId=
3375F
                      158350C}"
PassThroughClass    :
PassThroughIds      :
PassThroughNamespace :
PassThroughServer   :
UniqueId            : {D5F21992-E893-3B99-98A4-3375F158350C}
FriendlyName        : Masking set for EqualLogic volume CFSMPvol0
HostType            : Microsoft Windows
Name                : Masking set for EqualLogic volume CFSMPvol0
PSComputerName      :

Creating a volume  CFSMPvol1  in  default  on EqualLogic group  tekmktlab-10Gb
CFSMPvol1                                       OK                     Healthy
10 GB
Now adding the masking set to allow for this volume to be accessed to by Server: iqn.1991-
05.com.microsoft:cfscvmm2k12sp1.spartan.local  to  tekmktlab-10Gb

ObjectId            : {1}\\CFSCVMM2K12SP1\root/Microsoft/Windows/Storage/Providers\EQL_MaskingSet.ObjectId=
3DFF1
                      2F6D778}"
PassThroughClass    :
PassThroughIds      :
PassThroughNamespace :
PassThroughServer   :
```

You will now be able to use the volumes created.

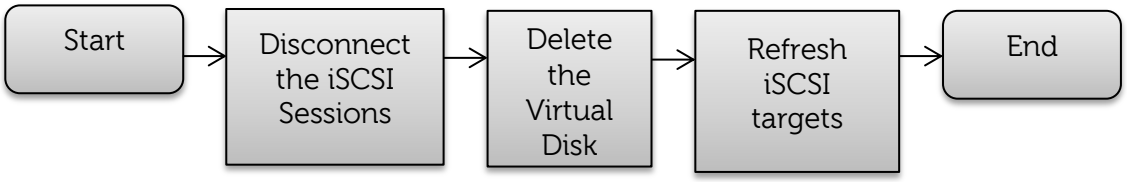
Removing volumes with native Windows Server 2012 cmdlets

This routine will allow you to quickly remove the volumes in bulk using the same iteration process as the Add volume with native Windows Server 2012 SM API cmdlets.

The following example requires Windows Server 2012 or Windows 8 and the Dell EqualLogic Host Integration Tools for Microsoft v4.5 or higher.

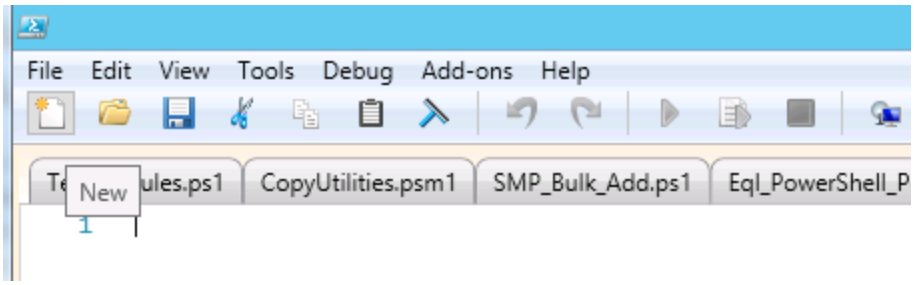
The steps to remove a volume are:

1. Disconnect the iSCSI connections to the EqualLogic volume.
2. Delete the virtual disk which will delete the EqualLogic volume.
3. Refresh the iSCSI Targets to reflect the change



Below is for Reference only and should only be used with careful consideration of the impact. Not all exceptions are captured.

In Windows Server 2012 PowerShell ISE create a new page



IMPORTANT: This PowerShell script as shown will delete 3 volumes from the server and PS Series array. Recommend the *$totvols* be set to 1 so only 1 volume is deleted until the script is fully understood.

Copy/Paste the code below with the appropriate variable changes into the new page:

```
#--------------------------------------------------------------------------
#  Copyright (c) 2011-2013 by Dell Inc.
#
# All rights reserved.  This software may not be copied, disclosed,
# transferred, or used except in accordance with a license granted
# by Dell Inc.  This software embodies proprietary information
# and trade secrets of Dell Inc.
#
#--------------------------------------------------------------------------

#####################################################
# Windows Server 2012/8 and EqualLogic HIT/ME v4.6   #
#####################################################
```

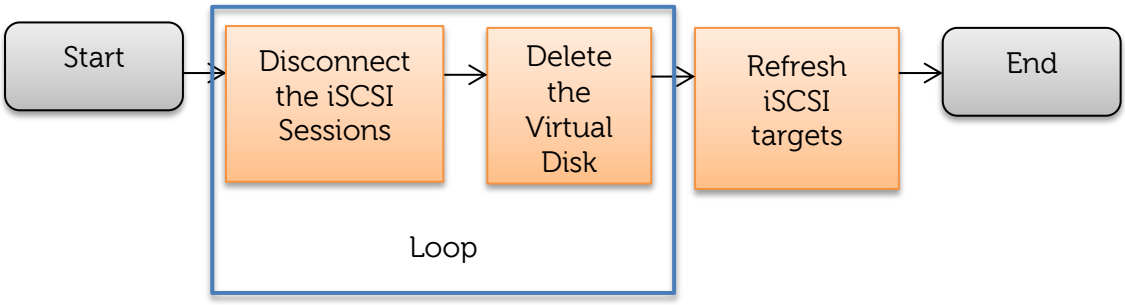
```
$myVol=""
$myTVol="SMPvol"
$myGroup ="tekmktlab-10Gb"
$totVols = 2
$myGroup ="tekmktlab-10Gb"
$initaddress=(get-initiatorport)
$tarport=(get-targetportal)

#Refresh the target
Get-iSCSITargetPortal | Update-iSCSITargetPortal

#Iterate through the volumes using the same pattern used to Create
for ($i=0;$i -le $totVols;$i++){
$myVol = $myTVol + $i
$iSCSITargetNodeAddr=(Get-VirtualDisk -FriendlyName $myVol | Get-
TargetPort).nodeaddress
Write-Host "Disconnecting" $initaddress.NodeAddress " to " $myVol

#Disconnect the volume
Disconnect-IscsiTarget -NodeAddress $iSCSITargetNodeAddr -Confirm:$false
Write-Host "Deleting " $initaddress.NodeAddress " to " $myVol " from " $myGroup
#Delete the volume from the PS Series array
Remove-VirtualDisk -FriendlyName $myVol -Confirm:$false
}
Write-Host "Updating all the iSCSI Targets this may take several minutes"
Update-IscsiTarget
```

Save the PowerShell script as "EQL_W2K12_BULK_DELETE.ps1" or other name as appropriate and execute through the ISE or at the command line.

To execute through the Windows PowerShell ISE click the green arrow  or from the Menu click Debug-> Run Continue. You may set breakpoints at any execution line for further interrogation.

Explanation of the script steps above:

Since Windows Server 2012/Windows 8 have fully integrated storage cmdlets the process to remove disks is simply a matter of disconnecting the iSCSI sessions with *Disconnect-iSCSITarget* and then removing the virtual disk with *Remove-VirtualDisk* and then finally completely refreshing all the targets with *Update-iSCSITarget*.



Example Output below:

```
PS C:\ > .\EQL_W2K12_BULK_DELETE.ps1
Disconnecting iqn.1991-05.com.microsoft:cfscvmm2k12sp1.spartan.local  to  CFSMPvol0
Deleting  iqn.1991-05.com.microsoft:cfscvmm2k12sp1.spartan.local  to  CFSMPvol0  from
tekmktlab-10Gb
Disconnecting iqn.1991-05.com.microsoft:cfscvmm2k12sp1.spartan.local  to  CFSMPvol1
Deleting  iqn.1991-05.com.microsoft:cfscvmm2k12sp1.spartan.local  to  CFSMPvol1  from
tekmktlab-10Gb
Disconnecting iqn.1991-05.com.microsoft:cfscvmm2k12sp1.spartan.local  to  CFSMPvol2
Deleting  iqn.1991-05.com.microsoft:cfscvmm2k12sp1.spartan.local  to  CFSMPvol2  from
tekmktlab-10Gb
```

Conclusion

Dell® EqualLogic Host Integration Tools for Microsoft® along with Auto-Snapshot Manager Microsoft® Edition (ASM/ME) provide robust integration and management capabilities for Dell PS Series storage. The exercises in this document should provide some guidelines on how Dell PS series storage environments may be best automated to improve management simplicity and optimization.

The references in this document to automate creation, deletion of Dell PS Series volumes and the automation of SAN Headquarters reporting only scratch the surface on the many capabilities from the suite of tools available to manage your enterprise.

Further capabilities include but are not limited to:

- Over 60 Dell EqualLogic PowerShell cmdlets,
- Auto-Snapshot Manager for Microsoft or Linux CLI
- Perl and Python scripts using PSAPI commands
- Windows Server 2012 Native PowerShell cmdlet automation
- Systems Center Virtual Machine Manager 2012 Native PowerShell cmdlet integration

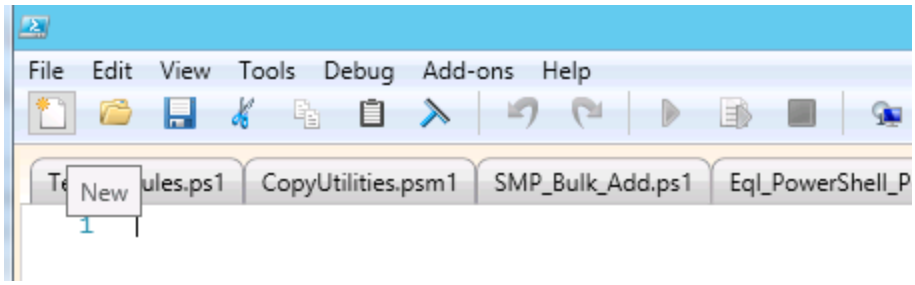
Note: references to "Native" simply indicate that the Windows Operating System PowerShell cmdlets work on Dell EqualLogic storage with the Dell Storage Management Provider.

Appendix A: Full Script to add a volume (Windows 2008)

This is the full script built from the step by step guide we introduced in the previous sections. This script may be run on Windows 2008 or Windows Server 2012/Windows 8.

Note: This script will need access to the PS Series Array Group and the volume must not exist.

Launch the Windows ISE and create a new PowerShell script (File -> New) and copy and paste the following. Please see the section on PS Series Group Access and the section to delete the volume if needed.



Copy/Paste with the appropriate variable changes into a new Windows ISE PowerShell Page

```
#-----------------------------------------------------------------------------
#  Copyright (c) 2011-2013 by Dell Inc.
#
# All rights reserved.  This software may not be copied, disclosed,
# transferred, or used except in accordance with a license granted
# by Dell Inc.  This software embodies proprietary information
# and trade secrets of Dell Inc.
#
#-----------------------------------------------------------------------------
################################################
# Windows 2008 or Windows 2012/8 compatible    #
################################################
#Steps:
#1. Create the volume in the appropriate pool
#2. Create the ACL To that volume
#3. Login to the iSCSI target (connects the sessions - MPIO)
#4. Format, Assign and bring the new volume online

#Start with constants
#Change these values to match your environment
#NOTE: These variables COULD BE READ IN FOR MORE Automation or incremented by
some pattern


$MinFree = 15000 #15GB free needs to be tested for min space...
$VolUsable = 10000 #size of volume
$PoolName = 'MyPool' #Please change to your pool
$GroupName= 'MyGroupName' #The group we initiated access.
$GroupLabel='GRPID'       #Lable created on volume with a reference to Group
$ThisVol = 'MyVol'        #case sensitive EQL volume
$TargetPortal= '127.0.0.1'  #Please change to the IP of your PS Series Array.


#Get the next drive letter
function Get-NextFreeDriveLetter
{
    return (Get-ChildItem function:[d-z]: -n | ?{ !(test-path $_) } | Get-
Random).Substring(0,1)
}
```

```
#List only pools with enough free space
#TECH NOTE:the following commands are PS Series cmdlets
#if the volume already exists a warning will be presented
$MyPools = Get-EqlPool|where-object  {$_.FreeSpaceMB -gt $MinFree}
foreach ($pool in $MyPools)
{
 [int]$FreeSpace = $pool.FreeSpaceMB/1024/1024
 [int]$FreeSapceMB =$pool.FreeSpaceMB

 write-host $pool.StoragePoolName Free:  $FreeSpace.ToString() TB

 #Lets just check for for enough free space -
 if ($pool.StoragePoolName -eq $PoolName){
    if ($FreeSapceMB -le $MinFree){
        Write-Host "This pool does not have enough free space " $PoolName
        exit 86
    }
   }
 }

################################################################################

#2.    Create the New PS Series Volume
##TECH NOTE:the following command is a PS Series cmdlet
 write-host "We will now create a new volume in " $PoolName " on PS Series
Group:" $GroupName
$newVolume = New-EqlVolume -VolumeName $ThisVol -VolumeSizeMB $VolUsable -
ThinProvision yes -StoragePoolName $PoolName -SnapshotBorrowingEnabled true -
GroupName $GroupName
if ($newVolume -le 0){
write-host $newVolume " Problem creating the volume"
}

#IQN and iSCSI Target will need to be retrieved before we can connect this host
################################################################################
#IQN is the iSCSI Qualified Name - the identifier for the Host
#This is also known as the initiator in iSCSI terminology
#The iSCSI target will represent the volume on the iSCSI SAN
################################################################################


#Windows 2003 and above
$hostn = ($env:computername)
#Next we need the iscsi initiator (host) for the iqn for new acl

#TECH NOTE:the following command is accessing the WMI api
$object   = get-wmiobject -namespace root\WMI -class
MSiSCSIInitiator_MethodClass -Computer $hostn
$IQN = $object.iSCSINodeName


#3.    Create the ACL from your host to the Volume for access. using the IQN from
above
Write-Host "New access enabled for " $ThisVol " to " $IQN " Host Name:" $hostn
#TECH NOTE:the following command is a PS Series cmdlet
new-EqlVolumeAcl -volumename $ThisVol -GroupName $GroupName -InitiatorName $IQN
-AclTargetType volume_and_snapshot

#Obtain the new iscsi target (the new volume)

#PS Series cmdlet to get the volume information specifically
#the iscsi target name
$MyVol = Get-EqlVolume -VolumeName $ThisVol -GroupName $GroupName

#Get the volume iscsi target name from volume information
$iTARGET=$MyVol.ISCSITargetName

#Refresh of the portal to update the initator
#TECH NOTE:the following command is using the Windows iscsicli command line
interface
Invoke-Expression ("iscsicli.exe refreshtargetportal $TargetPortal 3260")
```

```
#################################################################################
#Refresh the iscsi sessions
#TECH NOTE:the following commands use the WMI interface
#################################################################################
Write-Host "Now Connecting iscsi session to " $ThisVol " at this address "
$iTarget
$wmimthds = [wmiclass]"\\$hostn\root\wmi:MSiSCSIInitiator_MethodClass"
$wmimthds.RefreshTargetList #all sessions are refreshed
sleep(3) #sleep 30 seconds to let the refresh take place
$objTarget = get-wmiobject -computername $hostn -namespace root\wmi -class
MSiSCSIInitiator_TargetClass | where { $_.TargetName -eq $iTARGET }
$objLoginOpts = $null    # This could be filled in with authentication
                         # options. For this demo we will leave null.
##
# Login to the target - first is the normal login, second adds it
# as a persistent target so that volume will persist between server reboots
##
 $retn1 = $objTarget.Login($false, $null, $null, 0, $null, $objLoginOpts, 0,
$false)
 $retn2 = $objTarget.Login($false, $null, $null, 0, $null, $objLoginOpts, 0,
$true)

##
#Refresh after login
$wmimthds.RefreshTargetList
Invoke-Expression ("iscsicli.exe refreshtargetportal $TargetPortal 3260")
sleep(10) #Sleep 30 or so for refresh adjust time if needed
#Occasionally the refresh needs a little time to sync


#Windows Server 2012 has some alternatives to the iscsicli/wmi
#################################################################################
#Connect-IscsiTarget -NodeAddress $iTARGET
#Update-IscsiTarget -NodeAddress $iTARGET
#################################################################################

#Bring the volume online
#TECH Note: use the WMI interface to grab the target names for all iscsi
#sessions to our volume (multiple with MPIO)
#These sessions contain the Disk information so we know which disk to select
#for the DiskPart utility

#################################################################################
###Restart Point if the DiskNumber is over 1000...adjust sleep if needed
## simply rerun from here to the end###
#################################################################################
#  Grab the PhysicalDisk Objects for this iSCSI target
# - MPIO may have multiple sessions
#################################################################################
$colSessions = get-wmiobject -namespace root\wmi -class
MSiSCSIInitiator_SessionClass|where-object {$_.TargetName -eq $iTARGET}

#Method to weed out duplicates...
#use this list to see if we have seen the iscsi targetname already
#- need only one disk.
$targets     = @{}
foreach ($objSession in $colSessions)
    {


    # If we've already seen this target,
    # skip to the next one
    # Note:MPIO creates multiple sessions to the same device
    if($targets.ContainsKey($objSession.TargetName)) { continue }
    else                                             {
$targets.Add($objSession.TargetName, $true) }



    #Build a meaningful label matching the EQL Volume/Pool and Group
    # - helps with mapping resources between array and operating system
```

```
    $label = $MyVol.VolumeName + $PoolName + $GroupLabel

###############################################################################
    #Interrogate the devices property for the physical disk information

###############################################################################
    $colDevices = $objSession.Devices
    foreach ($objDevice in $colDevices)

        {
        #TECH Note: Windows DiskPart utility has to be executed to
        #manage the volume the Dispart utility will need the
        #disk number which will come from the  WMI iscsi initiator
        #session class to select and format since the utility
        #is not easy to invoke from powershell.
        #We will build the commands needed and pipe to the utility
        #
        #DiskPart operations:
        # 1. Select the newly created disk
        # 2. bring that disk online (ignore errors)
        # 3. Clean the disk
        # 4. Convert to a master boot record type for the volume
        # 5. Create a Partition on that disk
        # 6. Format and label the partition
        # 7. Assign the next available volume letter
        $letter= Get-NextFreeDriveLetter
        $diskID = $objDevice.DeviceNumber
        #creating the commands for the diskpart utility (no powershell
equivalent)
         Write-Host "Now initializing and formating (bringing online) PS Array
volume " $ThisVol " to Windows volume " $label " Disk#:" $diskID
        $dpscript = @"
        select disk $diskID
        online disk noerr
        ATTRIBUTES DISK CLEAR READONLY
        clean
        convert mbr
        create partition primary
        select part 1
        format fs=NTFS label=`"$label`" quick nowait noerr`n
        assign letter=`"$letter`"
"@
#the @ tags contain the commands...no whitespace allowed at the and @.
#pass (aka pipe) the commands as input to the diskpart utilitiy
    $dpscript | diskpart
    } #End of devices
}#End of sessions
Write-Host "You have completed the add volume script for " $ThisVol " to Windows
volume "$letter $label " Disk#:" $diskID
#End of Add volume PowerShell script for Windows 2008
```

To execute through the Windows PowerShell ISE click the green arrow  or from the Menu click Debug-> Run Continue. You may set breakpoints at any execution line for further interrogation.

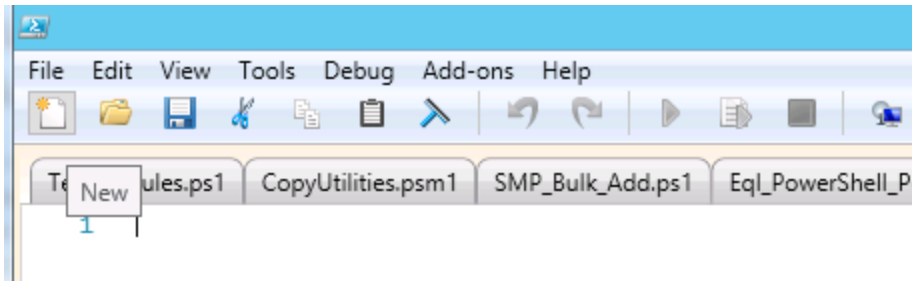
Example Output from the script

```
Execute either from the Windows ISE GUI or:
PS C:\> .\MyPath\MyAddVoltoPSA.ps1

Output:
PS C:\> .\EQLWin2008AddVolume.ps1

default Free: 23 TB
SSD-SAS Free: 10 TB
We will now create a new volume in  default  on PS Series Group: tekmktlab-10Gb
New access enabled for  TD1089V1  to  iqn.1991-05.com.microsoft:cfw2k12rcc2.spartan.local
Host Name: CFW2K12RCC2
PSAPI.Cmdlets.NewVolumeAcl TD1089V1 completed successfully.
Microsoft iSCSI Initiator Version 6.2 Build 8400

The operation completed successfully.
Now Connecting iscsi session to  TD1089V1  at this address  iqn.2001-05.com.equallogic:4-
52aed6-dc9080a73-f661c034d3b50f59-td1089v1
Microsoft iSCSI Initiator Version 6.2 Build 8400

The operation completed successfully.
Now initializing and formating (bringing online) PS Array volume  TD1089V1  to Windows
volume  td1089v1defaultTM10GB  Disk#: 12

Microsoft DiskPart version 6.2.8400

Copyright (C) 1999-2012 Microsoft Corporation.
On computer: CFW2K12RCC2

DISKPART>
Disk 12 is now the selected disk.

DISKPART>
DiskPart successfully onlined the selected disk.

DISKPART>
Disk attributes cleared successfully.

DISKPART>
DiskPart succeeded in cleaning the disk.

DISKPART>
DiskPart successfully converted the selected disk to MBR format.

DISKPART>
DiskPart succeeded in creating the specified partition.

DISKPART>
Partition 1 is now the selected partition.

DISKPART>


The format has been initiated successfully and is currently in progress.
Please wait until format has completed before trying to access the volume.

DISKPART>
DISKPART>
DiskPart successfully assigned the drive letter or mount point.

DISKPART>
```

Appendix B: Full Script to delete a volume (Windows 2008)

In the Windows ISE click the "New" script icon. Copy and paste the code below, and save with a meaningful name such as PSDeleteVolume.ps1.

In Windows PowerShell ISE create a new page



Copy/Paste below with appropriate variable changes into the new script page:

```
#--------------------------------------------------------------------------
#  Copyright (c) 2011-2013 by Dell Inc.
#
# All rights reserved.  This software may not be copied, disclosed,
# transferred, or used except in accordance with a license granted
# by Dell Inc.  This software embodies proprietary information
# and trade secrets of Dell Inc.
#
#--------------------------------------------------------------------------
#####################################################################
#Windows 2008 or Windows 2012/8 compatible                          #
#####################################################################
#Remove volume from the array works with the TechDoc Add Volume example
#
#Steps
#TAKE the Windows VOLUME OFFLINE with DiskPart
#Remove all iscsi sessions
#Take the PS Series volume offline
#Delete the PS Series volume
#####################################################################

#Only need to match the volume name created...all else is automated.

$ThisVol = 'MyVol'        #case sensitive EQL volume
$TargetPortal= '127.0.0.1'  #Please change to the IP of your PS Series Array.


## This comes from the iSCSI initiators GUI Discovery tab or ASM/ME PS Settings

$MyVol = Get-EqlVolume -VolumeName $ThisVol
$iTARGET=$MyVol.ISCSITargetName
#Take the one volume offline (filter by iscsi target name)
$colSessions = get-wmiobject -namespace root\wmi -class
MSiSCSIInitiator_SessionClass|where-object {$_.TargetName -eq $iTARGET}
#only need to disconnect the disk once
$targets     = @{}
foreach ($objSession in $colSessions)
{
    $colDevices = $objSession.Devices
    # If we've already seen this target, skip to the next one (MPIO creates
multiple sessions to the same device)
    if($targets.ContainsKey($objSession.TargetName)) { continue }
    else                                             {
$targets.Add($objSession.TargetName, $true) }

    foreach ($objDevice in $colDevices)
    {
```

```
    $disknum = $objDevice.DeviceNumber
    $diskID = $disknum
    $diskModel = $disk.model
    #creating the commands for the diskpart utility (no powershell equivalent)
    Write-Host "Will vary " $ThisVol " Disk#:" $disknum " Offline"
    $dpscript = @"
    select disk $diskID
    clean
    offline disk noerr
"@
    $dpscript | diskpart

    }
}

# Get any sessions associated with the target and log them out
#
$hostn = ($env:computername)
$colSessions = get-wmiobject -computername $hostn -namespace root\WMI -class
MSiSCSIInitiator_SessionClass | where { $_.TargetName -eq $iTARGET }

    foreach ($objSession in $colSessions)
    {
        write-host "Host $hostn - logging out ISCSI target $iTARGET"
        $retn = $objSession.Logout()
    }

    #
    # Remove the persistent connection as well
    #
    $colPersistent = get-wmiobject -computername $hostn -namespace root\WMI -
class MSiSCSIInitiator_PersistentLoginClass | where { $_.TargetName -eq $iTARGET
}

    foreach ($objPersistent in $colPersistent)
    {
        write-host "Host $hostn - removing persistent target:"
$objPersistent.TargetName
        $objPersistent.psbase.Delete()
    }



#TAKE PS Series VOLUME OFFLINE using PS Series Array PowerShell cmdlets

Write-Host "Now seting the PS Array volume " $ThisVol " to  Offline"
set-eqlvolume -volumename $ThisVol -OnlineStatus offline

Write-Host "Now removing PS Array volume " $ThisVol

#Remove the PS Series VOLUME using PS Series Array PowerShell cmdlets

Remove-EqlVolume -VolumeName $ThisVol -Force

Write-Host "Now refreshing the target."

#Refresh with the iscsicli.exe Windows utility

Invoke-Expression ("iscsicli.exe refreshtargetportal $TargetPortal 3260")

Write-Host "Successfully removed PS Array volume " $ThisVol

#End of Delete volume PowerShell script
```

To execute through the Windows PowerShell ISE click the green arrow  or from the Menu click Debug-> Run Continue. You may set breakpoints at any execution line for further interrogation.

Example Output:

```
PS C:\> .\PSDeleteVolume.ps1

Will vary  TD1089V1  Disk#: 12  Offline

Microsoft DiskPart version 6.2.8400

Copyright (C) 1999-2012 Microsoft Corporation.
On computer: CFW2K12RCC2

DISKPART>
Disk 12 is now the selected disk.

DISKPART>
DiskPart succeeded in cleaning the disk.

DISKPART>
DiskPart successfully offlined the selected disk.

DISKPART>
Host CFW2K12RCC2 - logging out ISCSI target iqn.2001-05.com.equallogic:4-52aed6-dc9080a73-
f661c034d3b50f59-td1089v1
Host CFW2K12RCC2 - logging out ISCSI target iqn.2001-05.com.equallogic:4-52aed6-dc9080a73-
f661c034d3b50f59-td1089v1
Host CFW2K12RCC2 - removing persistent target: iqn.2001-05.com.equallogic:4-52aed6-
dc9080a73-f661c034d3b50f59-td1089v1
Now seting the PS Array volume  TD1089V1  to  Offline
Volume 'TD1089V1' changed successfully
Now removing PS Array volume  TD1089V1
PSAPI.Cmdlets.RemoveVolume TD1089V1 completed successfully.
Now refreshing the target.
Microsoft iSCSI Initiator Version 6.2 Build 8400

The operation completed successfully.
```

Appendix C: Troubleshooting Techniques

Occasionally problems with scripting occurs which may be the result of different environments, sequence of events or other errors.

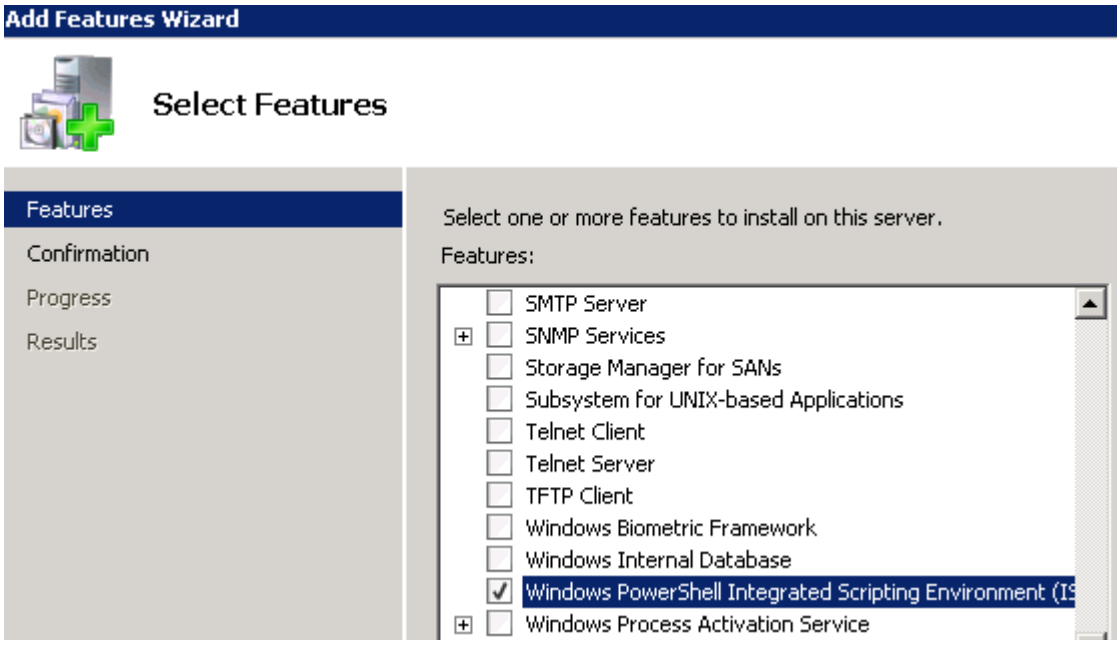
Below is a table to help identify and resolve issues with PowerShell scripting

Problem	Cause	Resolution
Issue with my Group access	Name or Group IP may be incorrect.	If this group had been previously connected you may need to re-register. "Remove-EqlGroupAccess –GroupName $GroupName" and then "New-EqlGroupAccess".
Null values	Typically something may be out of order and or did not initialize correctly.	First identify the reason for the null…i.e. cut and paste the code snippet in the cmd and then enter. PS C:\>$WhyIsThisNull <enter> This will show the value of any variable – once you identify the piece of code that did not work a resolution may be fairly easy to understand
Initialize-Disk : Cannot validate argument on parameter 'InputObject'. The argument is null. Supply a non-null argument and try the command again.	The iSCSI initiators may not be in the correct state. Connected already or not connected when they should be.	Update-iSCSITarget - may be necessary or in the iSCSI Initiator GUI you may refresh the targets and view to be sure they are in the correct state.
I am having a hard time understanding the code	Some code is a little more obvious than others	Use the ISE debugger to step though the code and see the results of any variable (anything with a $). Also the PowerShell ISE allows you to copy/paste any portion of your script into the command prompt with full edit capability. This is an excellent debugging technique to change

		values. The options allow you to start from a new location in the code or simply review variable assignments.
Volume already exists error	Deletion script for Windows Server 2012 or the one for Windows 2008 may not have been run or ran incorrectly.	Check the script for errors or look at the PS Series Group Manager GUI for existence of these volumes…if you intended have them deleted you will need to delete manually or through the appropriate scripts.
DISKPART> **The disk you specified is not valid.**	The iSCSI refresh occasionally will not complete before the DISKPART command starts. Additional information Now initializing and formatting (bringing online) PS Array volume CF2K8Power to Windows volume CF2K8Powerdefault Disk#: 4294967295	In the Windows 2008 Add volume script copy the lines starting above the $colSessions assignment to the end of the script Also consider increasing the sleep(X) routine to allow for the initialization to complete.

Appendix D: Windows 2008 Integrated Scripting Environment (ISE) setup

1. Click Start ->All Programs->Server Manager
2. Click on the "Features"
3. On the right click "Add Features"
4. In the "Add Features Wizard" shown below check the "Windows PowerShell Integrated Scripting Environment (ISE)"



Technical Support and Customer Service

Dell support service is available to answer your questions about PS Series SAN arrays.

Contacting Dell

1. If you have an Express Service Code, have it ready.
   The code helps the Dell automated support telephone system direct your call more efficiently.

2. If you are a customer in the United States or Canada in need of technical support,
   call 1-800-945-3355. If not, go to Step 3.

3. Visit support.dell.com/equallogic.

4. Log in, or click "Create Account" to request a new support account.

5. At the top right, click "Contact Us," and call the phone number or select the link for the type
   of support you need.