



Technical Report

NetApp PowerShell Toolkit

Best Practices Guide

Brahmanna Chowdary Kodavali and Shashanka SR, NetApp
October 2016 | TR-4475

Abstract

This technical report describes how to use the NetApp® PowerShell Toolkit (PSTK) in your environment to manage NetApp 7-Mode controllers, clustered Data ONTAP® controllers, and E-Series controllers. This report also includes instructions for using the most common commands and examples of how to integrate those controller-optimized commands with your host-based infrastructure.

TABLE OF CONTENTS

1	Introduction	5
1.1	Intended Audience	5
1.2	Scope of Document	5
1.3	Concepts	5
2	Technology Requirements	7
3	Installing PSTK	9
3.1	Installation Location	9
3.2	User Requirements	9
3.3	Commands List	9
3.4	Windows Remote Management Service Commands	9
4	PSTK Commands (7-Mode and Clustered Data ONTAP)	10
4.1	Finding and Licensing Controllers	11
4.2	Controller Commands	12
4.3	Common Host Commands	13
4.4	Creating and Configuring LUNs	19
4.5	Specific Volume Commands	22
4.6	Host-Based LUN Interrogation Commands	25
4.7	Aggregate and Disk Commands	28
4.8	Common SMB Commands	32
4.9	NFS Commands	39
4.10	Deduplication Commands	41
5	PowerShell Commands for SANtricity (E-Series)	43
6	Advanced User Permissions	52
7	Integration with Additional Windows Features	55
7.1	Event Logging	55
8	Useful Sample Scripts	57
8.1	AutoSupportChecker Script	57
8.2	IgroupChecker Script	58
8.3	SANKit	59
	Appendix: PSTK Module Release Notes	72
	Data ONTAP PSTK	72
	SANtricity PSTK	72

System Requirements	72
Installation Requirements	73
Support and Additional Information	73
Recommended Additional Reading	73
Version History	73

LIST OF TABLES

Table 1) Version of PowerShell that comes with each Windows OS version	8
Table 2) PSTK command categories covered in this technical report.	10
Table 3) PSTK command categories not covered in this technical report.	10
Table 4) Complete list of controller and credential commands.	12
Table 5) Common commands to connect an iSCSI LUN to the server.....	14
Table 6) FC target commands.....	17
Table 7) Complete list of igroup commands.	18
Table 8) Complete list of all LUN commands.	19
Table 9) Complete list of volume commands.....	23
Table 10) Default overrides.	29
Table 11) Complete list of aggregate commands.	29
Table 12) Complete list of disk commands.....	31
Table 13) Commands for joining CIFS service to active domain.	32
Table 14) Example commands for configuring SMB shares.....	33
Table 15) Valid option names.....	33
Table 16) Commands for domain management and returning NetBIOS names.	33
Table 17) Commands for CIFS share management.	34
Table 18) Commands for setting and modifying passwords.	35
Table 19) Commands for setting up and using ACLs and privileges to control access.	35
Table 20) Commands for setting permissions for users and groups on specific shares.	36
Table 21) Commands for CIFS sessions and statistics.	37
Table 22) Commands for VSS-based Snapshot copy management.	37
Table 23) Commands for home directory management.	38
Table 24) Commands for BranchCache services management.	38
Table 25) Commands for Symlink and character map management.....	39
Table 26) Commonly used configuration settings for an NFS service.....	39
Table 27) Commands for setting up and configuring an NFS service.	40
Table 28) Commands for exporting an NFS share.	40
Table 29) Commands for monitoring statistics for the NFS service.	41
Table 30) Commands for troubleshooting and fine-tuning an NFS export or NFS service.	41
Table 31) Complete list of deduplication commands.	42

Table 32) PSTK command categories for SANtricity controllers.	43
Table 33) API calls required to execute PowerShell commands related to controller licenses.	52
Table 34) Example of single API running multiple commands.	52
Table 35) Direct system control APIs.	53
Table 36) Wildcard-enabled permissions list.	54
Table 37) Suggested APIs for nonadmin role access on controller.	54
Table 38) SANKit error codes.	63
Table 39) SANKit code versions.	64
Table 40) General system requirements.	72
Table 41) Data ONTAP PSTK requirements.	73
Table 42) SANtricity PSTK requirements.	73

LIST OF FIGURES

Figure 1) GUI view of the output of the <code>out-gridview</code> command.	6
Figure 2) Flowchart showing the SANKit installation process.	60
Figure 3) Flowchart showing SANKit independent installation steps.	61
Figure 4) Flowchart showing decision matrix for SANKit.	62
Figure 5) Output from FCInfo command showing HBA information.	64
Figure 6) Output from FCInfo command showing details about installed HBAs.	65
Figure 7) Sample output from FCInfo showing detected NetApp target devices.	65
Figure 8) HBA firmware update (example).	66

1 Introduction

The NetApp PSTK integrates the following NetApp controllers with the Microsoft Windows OS to make them highly automatable and manageable:

- NetApp Data ONTAP operating in 7-Mode and clustered Data ONTAP controllers
- NetApp E-Series controllers (running NetApp SANtricity® software)

Microsoft is moving toward a more CLI-based interface to manage Windows servers. This trend started with Windows Core Server and is accelerating with the Windows Nano Server. The CLI-first approach does not preclude a user from managing an environment through a GUI; however, the GUI often lacks all of the options, flexibility, and power offered in the CLI environment. With this in mind, PSTK was written to expose all of the functions and features of the NetApp controllers using the NetApp Manageability Software Development Kit (SDK) open interface. These features include even those that are not available in our GUI management platforms.

1.1 Intended Audience

Experts in Microsoft PowerShell will find that all of the NetApp commands are self-documented, cross-referenced, and object-based. Therefore, PowerShell experts can immediately produce powerful and robust scripts. This document, however, is for individuals who are less familiar with PowerShell and its advantages when used to manage an infrastructure. The document is organized to help the reader understand PowerShell concepts while being able to quickly perform the most common, basic tasks used by a storage administrator. PowerShell has a gentle learning curve because it acts as both a CLI and a scripting language. You can use PowerShell as a simple Telnet/SSH replacement to perform basic tasks or feed the output of single commands into follow-up commands to produce custom operations that could not be done from a Telnet/SSH-type interface. As your abilities using PowerShell increase, you can add conditional statements such as `if/then/else` or loop-type statements such as `for-each`.

Note: These more advanced topics are optional; you are not required to adopt them if you choose not to.

1.2 Scope of Document

This document is not intended to make you an expert in programming concepts such as conditional statements (`if/then/else`), loops (`for-each/while`), or variables. If you plan to write complex PowerShell scripts for your infrastructure, NetApp recommends that you gain proficiency either by taking a basic PowerShell class or by reading a book on the subject. The appendix contains a list of books about gaining PowerShell proficiency.

1.3 Concepts

Objects Rather Than Output

The most valuable PowerShell concept is that you work with objects instead of text. For example, if you issue the `Get-NaLUN` command to return a list of LUNs on a NetApp storage controller, the command might return 10 LUNs or 12 LUNs. Each of these LUNs has a collection of data that describes it; this collection of data is not displayed on the screen by default. The command returns the LUNs and not the collection of data with which it is associated. This approach is different from that of a standard CLI for which the output is simply text that you can read. In the case of a standard CLI, you cannot automate decisions based on that output without first writing a parser to extract the meaning from the text.

Controlling Screen Output

The collection of LUNs is an object and the screen displays a view into that object (called a default format).

```
PS:> Get-NaLUN
Path                Size      SizeUsed  Protocol      Online Mapped
/vol/vol1/LUN7     100GB    34GB     windows_2008  True  True
/vol/vol1/LUN8     100GB    37GB     windows_2008  True  True
/vol/vol2/LUN3     100GB    33GB     windows_2008  True  True
```

In this example, the fields displayed are Path, Size, Size Used, Protocol, Online, and Mapped, but these are not all of the possible fields. To display all of the possible fields, run the following command:

```
PS:> Get-NCLUN | get-member
```

There are 62 types and numbers of fields for these objects, so they cannot all fit on the same screen. However, you can pick and choose which fields from the list you want to display by using the `format-list` option.

```
PS:> Get-NCLUN | format-list path,vserver,size,serialnumber
```

To insert all of these commands into a GUI so that you can manually experiment with different filters, run the following command:

```
PS:> Get-NCLUN | out-gridview
```

Figure 1 shows a sample of the command output when it was piped to the `out-gridview` command.

Figure 1) GUI view of the output of the `out-gridview` command.

Path	Size	SizeUsed	Protocol	Online	Mapped	Thin	Vserver
/vol/Win2012R2_BootGold_vol/FujiC4R42_BootLUN	50.0 GB	10.6 GB	windows_2008	True	True	True	infra
/vol/Win2012R2_BootGold_vol/FujiC4R43_BootLUN	150.0 GB	8.1 GB	windows_2008	True	True	True	infra
/vol/Win2012R2_BootGold_vol/FujiC4R44_BootLUN	200.0 GB	115.1 GB	windows_2008	True	True	True	infra
/vol/Win2012R2_BootGold_vol/FujiC5R28_BootLUN	100.0 GB	62.9 GB	windows_2008	True	True	True	infra
/vol/Win2012R2_BootGold_vol/FujiC5R36_BootLUN	50.0 GB	39.4 GB	windows_2008	True	True	True	infra
/vol/Win2012R2_BootGold_vol/FujiC5R37_BootLUN	100.0 GB	50.1 GB	windows_2008	True	True	True	infra
/vol/Win2012R2_BootGold_vol/FujiC5R41_BootLUN	50.0 GB	19.9 GB	windows_2008	True	True	True	infra
/vol/Win2012R2_BootGold_vol/FujiC5R42_BootLUN	50.0 GB	15.1 GB	windows_2008	True	True	True	infra

Using Pipe or Bar Symbol

The commands demonstrated in the section “Controlling Screen Output” use the pipe or bar symbol (|) to take the output from a command and input it to the next command. This process enables administrators to perform tasks that they normally could not from a Telnet, SSH, or serial-type CLI or that would be cumbersome using a GUI. For example, to convert all of the LUNs on a specific controller from fully provisioned to thin provisioned, run the following command for each LUN one at a time:

```
PS:> Set-NCLunSpaceReserved /vol/vol3/LUN8 -off
PS:> Set-NCLunSpaceReserved /vol/vol2/LUN1 -off
PS:> Set-NCLunSpaceReserved /vol/vol3/LUN9 -off
```

To convert all of the LUNs in the system at one time, run the following command:

```
PS:> Get-NCLUN | Set-NCLunSpaceReserved -off
```

Setting Filter in Pipeline

Thin provisioning a large number of LUNs might be more useful if you could set criteria for each LUN. For example, filter down the list of LUNs by using the pipeline with the `Where-Object` selector.

In the following example, the symbol `$_` refers to the object being piped, and the `.key` refers to the field used when selecting what output you want to see.

The following command retrieves a list of all of the LUNs:

```
PS:> Get-NCLUN
Path                Size      SizeUsed  Protocol      Online Mapped
/vol/vol1/LUN7     100GB    34GB     windows_2008  True  True
/vol/vol1/LUN8     200GB    37GB     windows_2008  True  True
/vol/vol1/LUN9     220GB    39GB     windows_2008  True  True
/vol/vol1/LUN10    240GB    45GB     windows_2008  True  True
/vol/vol12/LUN3    300GB    33GB     windows_2008  True  False
```

To filter the list to include only the LUNs with a size greater than 200GB, run the following command.

Note: `-gt` means greater than. The new list contains only the LUNs that meet the correct criteria.

```
PS:> Get-NCLUN | Where-Object {$_.size -gt 150gb}
Path                Size      SizeUsed  Protocol      Online Mapped
-----
/vol/vol1/LUN8     200GB    37GB     windows_2008  True  True
/vol/vol1/LUN9     220GB    39GB     windows_2008  True  False
/vol/vol1/LUN10    240GB    45GB     windows_2008  True  True
/vol/vol12/LUN3    300GB    33GB     windows_2008  True  False
```

You can pipe the output to another filter and refine the list. In this example, to filter out LUNs that are not currently mapped to a host, run the following command.

Note: For a Boolean [true/false] field, you don't need to compare it to anything.

```
PS:> Get-NCLUN | Where-Object {$_.size -gt 150gb} | Where-Object {$_.Mapped}
Path                Size      SizeUsed  Protocol      Online Mapped
-----
/vol/vol1/LUN8     200GB    37GB     windows_2008  True  True
/vol/vol1/LUN10    240GB    45GB     windows_2008  True  True
```

In this example, the list is filtered to show only the LUNs that you want to see. To send this output to the command that forces these LUNs to be thin provisioned, run the following command:

```
PS:> Get-NCLUN | Where-Object {$_.size -gt 150gb} | Where-Object {$_.Mapped} | Set-
NCLunSpaceReserved -off
```

Pipeline filtering is useful when trying to maintain infrastructures that are very large because you can modify hundreds of elements at one time. For example, using a GUI to change the NetApp Snapshot® copy schedules for 7,000 LUNs could take several weeks; if run from Telnet, it could take days. From PowerShell, this process can be accomplished in about five minutes, and most of that time is spent setting up the filters correctly.

2 Technology Requirements

Previous versions of PSTK used Windows .NET Framework 3.5.1 (.NET351) and PowerShell version 2 or later. Installing the .NET351 Framework is problematic on the newer Windows OS; therefore, PSTK version 3.3 and later must use .NET4. With the refresh of the framework, PSTK 3.3 and later require Windows PowerShell 3.0 or later to operate. Consequently, Windows XP, Windows 2003, and Windows Vista do not support PSTK version 3.3 and later. To use these older operating systems (OSs), use PSTK version 3.2 or earlier.

This requirement for PowerShell 3.0 also requires updates to the Windows Server 2008, Windows 7, and Windows Server 2008r2 machines to enable PowerShell 3.0 before installing PSTK.

By default, different versions of Windows OSs come with different versions of PowerShell. If needed, newer versions of PowerShell can be installed in an OS with some limitations and prerequisites. Table 1 shows which version of PowerShell comes with each OS version.

Table 1) Version of PowerShell that comes with each Windows OS version.

OS Version	Version Included with OS	Maximum Upgradable Version
Windows 10 and Windows Server 2016	PowerShell 5.0	PowerShell 5.0
Windows 8.1 and Windows Server 2012r2	PowerShell 4.0	PowerShell 5.0
Windows Server 2012r2	PowerShell 4.0	PowerShell 4.0
Windows 8.0	PowerShell 3.0	PowerShell 3.0 (upgrade to 8.1)
Windows Server 2012	PowerShell 3.0	PowerShell 4.0 ^{1, 3}
Windows 7.0 and Windows Server 2008r2	PowerShell 2.0	PowerShell 4.0 ^{1, 2}
Windows Vista	PowerShell 1.0	PowerShell 2.0
Windows Server 2008	PowerShell 1.0	PowerShell 3.0 ²
Windows XP and Windows Server 2003r2	None	PowerShell 2.0

¹.NET4.5 must be installed.

²Service pack (Windows 7, Windows 2012r2, Windows 2008) must be installed.

³Windows Management Framework 4.0 (Windows6.1-KB2819745-x64-MultiPkg.msu) must be installed.

To verify which version of the PowerShell environment exists on your server, run the following PowerShell command:

```
PS:> $PSVersionTable
```

After you install PSTK, you can gain access to the commands and determine the installed version by opening a PowerShell window and running the following commands:

```
PS:> import-module dataontap
PS:> get-module dataontap | get-version
```

The newly installed version should show as a major version of 3, a minor version of 3, a build number of 0, and a revision number of 0.

By looking at all of the properties of the NetApp modules, you can also determine which versions of PowerShell are required for each of the controller types. Gather this information by running the following command:

```
PS:> get-module dataontap | format-list *
PS:> get-module netapp.santricity.powershell | format-list *
```

PowerShell 3.0 is the required version for each of the preceding modules. In many cases, a new PSTK version is installed on top of an existing version. NetApp recommends that you verify whether the new version of PSTK successfully overwrote the existing version.

3 Installing PSTK

3.1 Installation Location

For versions of PSTK earlier than 3.3, the toolkit module is commonly installed to the following location:

```
C:\Windows\System32\WindowsPowerShell\v1.0\Modules\DataONTAP
```

For PSTK version 3.3 and later, NetApp recommends that you install PSTK to the following location:

```
C:\Program Files (x86)\NetApp\Data ONTAP PowerShell Toolkit\DataONTAP
```

Update the environment variable that lets PowerShell find this location before it looks in the standard modules directory. The environment variable is called `$env:PSModulePath`. You can check the value of this variable by running the following command at a PowerShell prompt:

```
PS:> $env:PSModulePath
```

This location enables a system administrator to quickly determine which NetApp software is installed on a machine with a simple glance at this directory and without needing to know the default PowerShell modules directory.

This ability is beneficial when installing a newer version of PSTK and the old version still exists in the default module directory. If you run the `Import-Module` command to load the module from the modules directory, it might install the older version of PSTK instead of the new version.

Note: Using the method described in the previous section, validate that the correct version is installed.

3.2 User Requirements

PSTK can be installed by any user. After PSTK is installed, the module is available to any user. The module is self-contained. Therefore, the only thing needed for it to operate correctly is for the `$PSModulePath` variable to point to the NetApp installation directory inside the Program Files directory before it points to the PowerShell modules directory.

3.3 Commands List

To obtain a list of commands in each of the three modules, run the following PowerShell commands:

```
PS:> (Get-Module NetApp.SANTricity.PowerShell).exportedCommands  
PS:> (Get-Module DataONTAP).exportedCommands
```

The list of commands is too extensive to cover in a best practices guide; therefore, this guide covers the commonly used commands and the most recently added commands.

3.4 Windows Remote Management Service Commands

Using Windows Remote Management Service (WinRM), you can issue PowerShell commands to other computers in your infrastructure. This capability is useful to determine how LUNs or shares are deployed or to determine configuration settings for individual servers. You must first enable and configure WinRM. To verify the current settings of the WinRM service, run the following command:

```
PS:> WinRM get WinRM/Config
```

For the purposes of this document, it is assumed that the quick configuration settings can be used.

Note: For all of the custom settings that WinRM supports, see the Microsoft documentation.

```
PS:> WinRM quickconfig  
PS:> Enable-PSRemoting -Force
```

After the remote PowerShell is set up, set the list of acceptable hosts by running the following PowerShell command.

Note: In this example, the wildcard is used to allow all servers to issue PowerShell commands to this computer.

```
PS:.\> set-item wsman:\localhost\client\trustedhost *
PS:> restart-service WinRM
```

You can now issue PowerShell commands from a central computer to enabled machines. To issue individual commands or short collections of commands, run the simpler `Invoke-type` command. However, for an interactive session, use the `PSSession-type` command. See the following examples:

```
PS:> Invoke-Command -computername XYZ -scriptblock { get-NAHostDisk } -credential Chris
PS:> Enter-PSSession -Computername XYZ -credential Chris
[XYZ] PS:> Get-NAHostDisk # this command is run interactively
[XYZ] PS:> Get-NAiSCSIInitiator # this command is ALSO run interactively
```

4 PSTK Commands (7-Mode and Clustered Data ONTAP)

The commands in this section enable connection to an ONTAP controller. To connect to the controller, you must provide controller credentials.

More than 2,000 commands exist in PSTK that manage 7-Mode or clustered Data ONTAP controllers and hosted storage. These commands are organized into categories. This document covers the most commonly used categories and common commands in each category. The PSTK command categories shown in Table 2 are covered in this technical report.

Table 2) PSTK command categories covered in this technical report.

Command Categories Covered		
Aggr	FCP	Perf
CIFS	Host	NetApp SnapMirror®
Clone	Igroup	Snapshot
Copy offload	iSCSI	Toolkit
Disk	LUN	User admin
FC	NFS	Volume

The PSTK command categories shown in Table 3 are also available in the toolkit, but they are not covered in this technical report.

Table 3) PSTK command categories not covered in this technical report.

Command Categories Not Covered		
Active Directory	Group mapping	Security session
Antivirus	Kerberos	Security SSL
Audit	Job	Service processor
NetApp AutoSupport®	Lock	SES
CF	LDAP	NetApp SnapLock®

Command Categories Not Covered		
Clock	License	NetApp SnapVault®
Cluster	NetApp MetroCluster™	SIS
Cluster image	Name service	SnapMirror policy
Cluster peer	Name mapping	SNMP
Config backup	NDMP	Storage adapter
Consistency group	Net	Storage array
Core dump	Netgroup	Storage bridge
Dashboard	NIS	Storage initiator
Diagnosis	Ntp_server	Storage-iSCSI-initiator
Disk encrypt	Options	Storage pool
EMS	Portset	Storage port
Environment sensor	Priority	Storage switch
Exports	QoS	System
Feature	Qtree	UCM
FC port	Quota	NetApp vFiler®
File	Radius	Virtual machine (VM)
File directory security	Reallocate	VM services
File service audit	RSH	Vscan
Flash	Sectrace	Vserver
NetApp FlexCache®	Securadmin	Vserver peer
NetApp FPolicy®	Security	Vserver peer transition
HA interconnect	Security certificate	Volume Shadow Copy Service (VSS)
GPO	Security key manager	NetApp WAFL®

Some categories (such as EMS) have a single command while other categories (such as CIFS) have over 20 commands.

4.1 Finding and Licensing Controllers

By using PSTK, it is possible to find NetApp controllers within a subnet. The command assumes that you know both the subnet you want to scan and the subnet mask in prefix-length format. For example, a subnet mask of 255.255.255.0 would be represented as \24.

```
PS:> Find-NAController 10.20.30.0\24
PS:> Find-NCController 10.20.30.0\24
```

4.2 Controller Commands

The commands in this section are used to connect to ONTAP controllers (you are prompted for a password). The first command connects to a 7-Mode controller, the second command connects to an ONTAP cluster, and the third command connects to a storage virtual machine (SVM).

Note: Green console text highlights comments that explain each command on the line on which it appears.

```
PS:> Connect-NAController 10.20.30.40 -cred root # For a 7-Mode Controller
PS:> Connect-NCController 10.20.30.40 -cred admin # For a Clustered Data ONTAP Cluster
PS:> Connect-NCController 10.20.30.40 -cred vsadmin # For a Clustered Data ONTAP SVM
```

Note: Although it is possible to insert PowerShell commands that allow you to embed passwords and automate the connection to the controller, NetApp does not recommend doing so. The reason is because it leaves your passwords in clear text within the files.

Run the following embedded script to automate the connection to the controller:

```
PS:> $pass=ConvertTo-SecureString "password" -AsPlainText -Force
PS:> $cred=New-Object -TypeName System.Management.Automation.PSCredential -Argumentlist
"root",$pass
PS:> Connect-NAController 10.20.30.40 -cred $cred
```

Controller and Credential Commands

NetApp recommends that you run commands to create a credential store under the user who has access to the controller. This credential store allows the `Connect-NAController` and `Connect-NCController` commands to run without user name or password prompts. To add NetApp controller credentials to the current user, run the following command (you are prompted for a password):

```
Add-NACredetial 10.20.30.40 -cred root #for a 7-Mode Controller
Add-NCCredetial 10.20.30.40 -cred admin # for a Clustered Data ONTAP Cluster
Add-NCCredetial 10.20.30.40 -cred vsadmin # for a Clustered Data ONTAP SVM
```

Table 4 is a complete list of the controller and credential commands. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 4) Complete list of controller and credential commands.

Command	Function	Available In
Add-N?Credential	Saves login credentials for the ONTAP controller	7, C
Connect-N?Controller	Establishes a connection to an ONTAP storage controller	7, C
Dismount-NaController	Removes one or more PSDrive objects created by the ONTAP PowerShell provider	7
Find-N?Controller	Discovers the ONTAP controllers	7, C
Get-N?Command	Gets a list of ONTAP cmdlets	7, C
Get-NaControllerError	Gets a description of an ONTAP error code	7
Get-N?Credential	Lists entries in the credentials cache	7, C

Command	Function	Available In
Get-N?Efficiency	Gets efficiency information for a set of aggregates or volumes	7, C
Get-N?Help	Lists cmdlets in the ONTAP module, filtered by API, category, or cmdlet name	7, C
Get-NaHelpAlias	Lists aliases for cmdlet parameters	7
Get-NaHelpUnsupported	Lists cmdlets that rely on APIs that are not supported by an ONTAP controller	7
Get-NaToolkitConfiguration	Gets the basic configuration of PSTK	7
Get-NaToolkitVersion	Returns the executing version of PSTK	7
Initialize-NaController	Initializes a new ONTAP controller	7
Initialize-NcObjectProperty	Sets one or more properties on toolkit objects	C
Invoke-N?Perfstat	Gathers perfstat information	7, C
Invoke-NcServiceProcessorAutoSupport	Invokes AutoSupport from a specific service processor	C
Invoke-N?Ssh	Sends an ONTAP CLI command by using SSH	7, C
Mount-NaController	Mounts an NaController object as a PSDrive using the ONTAP PowerShell provider	7
Remove-N?Credential	Removes saved login credentials for the ONTAP controller	7, C
Set-NaToolkitConfiguration	Sets the basic PSTK configuration	7
Show-NcHelp	Displays PSTK documentation in a browser	7, C

4.3 Common Host Commands

Connect to the NetApp controller where you will manage the Fibre Channel (FC), iSCSI, NFS, or SMB storage types. Each of these storage types is described in the following sections.

Common Commands for iSCSI

This section describes the most common commands used to connect an iSCSI LUN to a server or a set of servers using PSTK. Although there are a large number of commands to manage and manipulate iSCSI, the commands described in this section are used to deploy a LUN.

Note: You need to know the iSCSI qualified name (IQN) of the host and the IP addresses of the NetApp iSCSI target to connect a host to a NetApp controller using iSCSI.

From the host, a common command set enables iSCSI on a host and then automates connecting the iSCSI host to the NetApp controller.

1. Run the following commands to start the iSCSI service on the host and then retrieve the IQN of the server (7-Mode and clustered):

```
PS:> Start-Service -name msiscsi -startuptype Automatic
PS:> Get-NCHostiSCSIAdapter
PS:> $MyIQN = (Get-NCGHostiSCSIAdapater).iqn # To cast it into a variable
```

2. Connect the host to an iSCSI target by completing the following steps:
 - a. Create a new iSCSI target portal to represent the NetApp iSCSI target SVM.

```
PS:> Add-NCHostiSCSITargetPortal
PS:> Add-NAHostiSCSITargetPortal
```

- b. Determine the IQN for the SVM.

```
PS:> Get-NCiSCSINodeName
```

- c. Determine the IP addresses of the iSCSI data LIFs on the clustered Data ONTAP controller.

```
PS:> Get-NCNetInterface | Where {$_.DataProtocols -contains "iscsi"}
```

3. Run the following command on each of these IP addresses to connect the multiple paths to the controllers from the host. Use the IQN for the SVM instead of the variable \$SVMIQN and use the IP address to each iSCSI data LIF instead of the variable \$DataLIF.

Note: Run this command for each data LIF to which you are connecting.

```
PS:> Connect-iSCSITarget -nodeaddress $SVMIQN -IsMultiPathEnabled $true -IsPersistent $True -
TargetPortalAddress $DataLIF
```

Table 5 shows common commands used to connect an iSCSI LUN to a server. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 5) Common commands to connect an iSCSI LUN to the server.

Command	Function	Available In
Add-NclscsInterfaceAccess	Adds the iSCSI LIF to the access list of the specified initiator	C
Add-NalscsilInitiatorAuth	Adds the initiator to the authentication list	7
Add-NclscsiService	Adds an iSCSI service in an SVM; each SVM requires an online iSCSI service to serve data using the iSCSI protocol	C
Add-N?IscsilInterfaceAccess	Adds the named network interfaces to the access list for the specified initiator	7, C
Add-N?IscsiTargetPortalGroupInterface	Adds an interface to an iSCSI target portal group	7, C
Clear-NalscsiStatistics	Zeros all iSCSI statistics counters	7
Disable-N?Iscsi	Stops iSCSI service	7, C
Disable-N?IscsilInterface	Disables one or more interfaces for use by iSCSI	7, C
Disable-N?Isns	Stops iSNS service	7, C
Enable-N?Iscsi	Starts iSCSI service	7, C
Enable-N?IscsilInterface	Enables one or more interfaces for use by iSCSI	7, C

Command	Function	Available In
Enable-N?Isns	Starts iSNS service	7, C
Get-N?IscsiConnection	Lists all iSCSI connections	7, C
Get-N?IscsiInitiator	Lists the initiators that are logged in	7, C
Get-N?IscsiInitiatorAuth	Gets the authentication information for an initiator	7, C
Get-N?IscsiInitiatorDefaultAuth	Gets the default auth information for iSCSI	7
Get-N?IscsiInterface	Lists the interfaces and their status for iSCSI	7, C
Get-N?IscsiInterfaceAccess	Lists the iSCSI interface access list objects	C
Get-N?IscsiNodeName	Returns the current iSCSI node name	7, C
Get-N?IscsiService	Lists the iSCSI services	C
Get-N?IscsiPortal	Lists the iSCSI portals	7
Get-N?IscsiSession	Lists the active iSCSI sessions	7, C
Get-N?IscsiStatistics	Gets iSCSI statistics for the current controller	7
Get-N?IscsiTargetAlias	Returns the current iSCSI target alias	7, C
Get-N?IscsiTargetPortalGroup	Lists information about iSCSI target portal groups	7, C
Get-N?Isns	Gets iSNS service configuration	7, C
New-N?IscsiChapPassword	Generates a 128-bit random password that can be used as a CHAP secret	7, C
New-N?IscsiTargetPortalGroup	Creates a new iSCSI target portal group	7, C
Remove-N?IscsiInitiatorAuth	Deletes the initiator from the authentication list	7, C
Remove-N?IscsiInterfaceAccess	Removes the named network interfaces from the access list for the specified initiator	7, C
Remove-N?IscsiService	Removes the iSCSI service on the SVM	C
Remove-N?IscsiTargetPortalGroup	Destroys an iSCSI target portal group	7, C
Remove-N?IscsiTargetPortalGroupInterface	Deletes an interface from an iSCSI target portal group	7, C
Remove-N?IsnsService	Destroys the iSNS service for an SVM	C
Set-N?IscsiInitiatorAuth	Sets the initiator authentication method	C
Set-N?IscsiInitiatorAuthChap	Modifies CHAP parameters of an existing per-initiator authentication record whose authentication type equals CHAP	7, C
Set-N?IscsiInitiatorDefaultAuth	Configures the default iSCSI authentication method	7
Set-N?IscsiNodeName	Sets the current iSCSI node name	7, C
Set-N?IscsiTargetAlias	Sets or clears the current iSCSI target alias	7, C

Command	Function	Available In
Set-N?IscsiTargetPortalGroupAlua	Changes the Asymmetric Logical Unit Access (ALUA) parameters on an iSCSI target portal group	7, C
Set-N?IscsiService	Modifies the iSCSI service in an SVM	C
Set-N?IsnsAddress	Configures the iSNS service	7, C
Test-N?Iscsi	Gets the status of the iSCSI service, whether or not it is running	7, C
Update-N?Isns	Forces iSNS service to update server	7, C

- To complete the setup of the host, proceed to the section “Common Commands for Igroup to FC/iSCSI/FCoE.”

Common Commands for FC/Fibre Channel over Ethernet

FC connections rely on HBAs, less on the host and array-side configuration than on the zoning operations, to expose LUNs to a host. To support these zoning commands, gather the worldwide port names (WWPNs) from the host as well as the SVM.

This document does not cover the FC switch commands or configurations because different vendors use different commands and these commands are unique to your environment.

To gather the WWPNs used by your SVM, run the following command:

```
PS:> Get-NCFCPInterface | Where {$_.Vserver -contains "SVMName"}
```

There is not a method to easily gather the WWPN from the host. The location in Windows Management Instrumentation (WMI) to store the worldwide name (WWN) information is designated as the worldwide node name (WWNN), which should not be used to zone an HBA. Download the HBA tools (CLI and GUI) for your specific brand HBA to gather this information. Alternately, use the WWNN, which can be viewed from the switch, to correlate the correct WWPN.

To gather the WWNN used by your host, run the following command:

```
PS:> Get-NCHostFCAdapter
```

After gathering the necessary information to zone the FC switches, verify that the WWNN for the host can be seen from the NetApp FCP target ports. HBAs log in to each target that they see, so it is possible to interrogate the NetApp controller to FCP to verify that the zoning on the switch is correct.

To determine the relationship between WWPN and WWNN connected to the NetApp controllers, run the following short script.

Note: Normally, a script is a file that you save and then execute later, but in PowerShell the distinction between a script and a command line doesn't exist. You can copy a complex script to a command line and it executes in the same way. In this example, we are executing a `ForEach`-type loop in a single line of text.

```
PS:> ForEach($X in GetNCFCPInitiator) {Write-Host "HBAs on Port "$x.adapter;
$x.FCPCConnectedInitiators}
```

A `ForEach` loop is used in this example to explore detailed information in each of the returned objects, but this command returns a collection of objects instead of a single object. This `ForEach` loop allows you to explore each object in the collection one by one. After you verify a valid connection from the host HBA to the NetApp FC target, continue to the section “Common Commands for Igroup to FC/iSCSI/FCoE” to deploy LUNs to this host.

Table 6 shows a complete list of FC target commands. In this table, 7 represents ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 6) FC target commands.

Command	Function	Available In
Disable-N?FcAdapter	Calls on the corresponding adapter driver disable function to bring the adapter offline	7, C
Enable-N?FcAdapter	Calls on the corresponding adapter driver enable function to bring the adapter online	7, C
Get-N?FcAdapter	Lists the FC adapters	7, C
Set-N?FcAdapterType	Changes the adapter driver and/or configuration state	7, C
Add-NcFcpService	Creates an FCP service in an SVM; each SVM requires an online FCP service to serve data using FCP	C
Clear-NaFcpAdapterStatistics	Resets the stats for the specified FC target adapter	7
Clear-NaFcpPortNameAlias	Removes an alias for a WWPN of an initiator	7, C
Disable-N?Fcp	Stops the FCP service	7, C
Disable-N?FcpAdapter	Brings the adapter offline	7, C
Enable-N?Fcp	Starts the FCP service	7, C
Enable-N?FcpAdapter	Brings the adapter online	7, C
Get-N?FcpAdapter	Gets information, such as node name/port name and link state, about the specified FC target adapter; or, if an adapter is not specified, gets information about all the FC target adapters	7, C
Get-N?FcpAdapterStatistics	Gets statistics about the FC target adapters	7, C
Get-NaFcpClusterMode	Gets the current cfmode setting for the system	7
Get-N?FcpInitiator	Gets the list of initiators currently connected to the specified FC target adapter	7, C
Get-N?FcpNodeName	Gets the current FCP node name	7, C
Get-N?FcpPortName	Lists the valid FC target port names on a storage system's local adapters	7, C
Get-N?FcpPortNameAlias	Lists the WWPNs for a given alias, an alias for a given WWPN, or a complete list of all current alias WWPN mappings	7, C
Get-NcFcpService	Lists the FCP services	C
Ping-NaFcp	Sends an ELS ECHO frame to an FC address or WWPN, returns true if the ping is successful	7
Ping-NaFcpInfo	Sends an ELS ECHO frame to an FC address or WWPN; the cmdlet pings the address or WWPN count times and returns the number of successful pings and times	7

Command	Function	Available In
Remove-NcFcpService	Destroys the FCP service in an SVM	C
Set-NaFcpAdapterMediaType	Sets the link type on the FC target adapter	7
Set-NaFcpAdapterPartner	Sets the name of the partner adapter that the local adapter should take over	7
Set-N?FcpAdapterSpeed	Sets the speed on the FC target adapter	7, C
Set-NaFcpClusterMode	Sets the current cfmode setting for the system	7
Set-N?FcpNodeName	Sets the current FCP node name	7, C
Set-N?FcpPortName	Sets a valid but unused port name on a local FC target adapter	7, C
Set-N?FcpPortNameAlias	Sets an alias for a WWPN of an initiator that might log in to the target	7, C
Switch-NaFcpPortName	Swaps the port names of two local FC target adapters	7
Test-N?Fcp	Gets the status of the FCP service, whether or not it is running	7, C

Common Commands for Igroup to FC/iSCSI/FCoE

An igroup is a relationship between a label (in most cases a host name) and a set of IQNs or WWPNs that allows either FC or iSCSI LUNs to be deployed to servers.

A common best practice is to use the NetBIOS or host name to create the igroup on the controller. If the host uses both iSCSI and FC, then NetApp recommends assigning a postfix to the name of the protocol. For example:

- Igroup name = SQLHost4_<iSCSI or FC>
- Igroup name = ExchHost3_iSCSI (is an iSCSI igroup if the host uses iSCSI and FC)
- Igroup name = SHPTHost2_FC (is an FC igroup if the host uses iSCSI and FC)

The methods to create new igroups for each type are described in this section. You need to know the target OS, the protocol choice, and the host name of the server.

Note: The `$Hostname` variable is the host name and the `$protocol` variable is either iSCSI or FCP.

```
PS:> New-NCiGroup $Hostname -Protocol windows
```

After the igroup is created, set specific settings such as adding a WWPN or an IQN. For example:

```
PS:> Add-NCiGroupInitiator $Hostname $IQN
PS:> Add-NCiGroupInitiator $Hostname $WWPN
```

The process of assigning a LUN to a specific igroup is the final step in the LUN creation process and is covered by the LUN creation and configuration commands. Table 7 shows a complete list of the igroup commands. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 7) Complete list of igroup commands.

Command	Function	Available In
Add-N?IgroupInitiator	Adds initiator to an existing initiator group	7, C

Command	Function	Available In
Add-N?IgroupPortset	Binds an existing igroup to a given portset	7, C
Get-NclgroupOSTypes	Displays the supported values and descriptions for initiators by OS group type	C
Get-N?Igroup	Gets information for initiator groups	7, C
Get-N?LunByIgroup	Finds the path to the LUN mapped at a given LUN ID for a given initiator group	7, C
New-N?Igroup	Creates a new initiator group	7, C
Remove-N?Igroup	Destroys an existing initiator group	7, C
Remove-N?IgroupInitiator	Removes nodes from an initiator group	7, C
Remove-N?IgroupPortset	Unbinds an existing igroup from a portset	7, C
Rename-N?Igroup	Renames an existing initiator group	7, C
Set-N?Igroup	Sets an attribute for an initiator group	7, C

4.4 Creating and Configuring LUNs

Creating LUNs allows storage to be assigned to various servers throughout an infrastructure. Creating a LUN requires a NetApp FlexVol® volume with the appropriate amount of space, a name for the LUN, and knowledge about the type of OS the LUN is deployed on and the size of the LUN.

1. Create a new LUN of the type Windows 2008 or later:

```
PS:> New-NALUN /vol/vol131/LUN2 100g -type windows_2008
```

2. To set thin provisioning to either on or off (now or at a later time), run the following command against the LUN:

```
PS:> Set-NCLUNSpaceReserved /vol/vol131/LUN2 -off # (or -on)
```

3. After a LUN is created, map it to a host using an igroup. Run the following command to add a LUN to or remove a LUN from an igroup:

```
PS:> Add-NCLUNMap /vol/vol131/LUN2 igroupname1
PS:> Remove-NCLUNMap /vol/vol131/LUN2 igroupname1
```

4. Run the following command to obtain a list of all of the igroups that are currently mapped to a specific LUN.

Note: This is especially useful for Windows cluster troubleshooting.

```
PS:> Get-NCLUNMap /vol/vol131/LUN2
```

Table 8 shows a complete list of all the possible LUN commands. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Note: The 7-Mode version of each command uses the NA prefix for the noun while the clustered Data ONTAP version of the command uses the NC prefix for the noun.

Table 8) Complete list of all LUN commands.

Command	Function	Available In
Add-N?LunMap	Maps the LUN to all the initiators in the specified initiator group	7, C

Command	Function	Available In
Add-NcLunMapReportingNodes	Adds nodes to the list reporting the LUN map	C
Clear-N?LunPersistentReservation	Clears the persistent reservation information for a given LUN	7, C
Clear-NaLunStatistics	Resets (zero) block protocol access statistics for LUNs	7
Confirm-N?LunHasScsiReservation	Queries for all types of SCSI reservations covering both iSCSI and FCP	7, C
Confirm-N?LunInitiatorLoggedIn	Determines if an initiator is logged in using FCP or iSCSI	7, C
Confirm-NaLunRestore	Gets the state of LUN restore	7
Get-N?Lun	Gets the status of the given LUN or all LUNs	7, C
Get-NcLunAlignment	Gets a list of LUN alignment objects	C
Get-N?LunAttribute	Gets a named attribute for a given LUN	7, C
Get-NaLunClone	Gets all LUN clones contained in a given Snapshot copy	7
Get-NaLunCloneSplit	Gets the cloning status of LUNs	7
Get-NcLunCopy	Gets details of LUN copy jobs on the controller	C
Get-NaLunComment	Gets the optional descriptive comment for a LUN	7
Get-N?LunGeometry	Gets SCSI disk geometry for a given LUN	7, C
Get-NcLunImport	Gets details of LUN relationships on the controller	C
Get-NaLunInquiry	Gets the SCSI inquiry response data for vendor ID (vid), product ID (pid), and firmware revision (rev) based on the igroup to which the LUN in question is mapped	7, C
Get-N?LunMap	Gets a list of initiator groups and their members (the initiators) mapped to the given LUN	7, C
Get-N?LunMapByInitiator	Lists all the LUNs mapped to an initiator	7, C
Get-N?LunMaxSize	Gets the maximum possible size, in bytes, of a LUN on a given volume or qtree	7, C
Get-N?LunMinSize	Gets the minimum possible size, in bytes	7, C
Get-NcLunMove	Gets details of LUN move jobs on the controller	C
Get-NcLunOsTypes	Displays the supported values and descriptions for LUN OS type	C
Get-NaLunOccupiedSize	Gets the size occupied by the LUN in the active FS	7
Get-N?LunPersistentReservation	Gets the persistent reservation information for a given LUN	7, C

Command	Function	Available In
Get-N?LunSelect	Gets the select attribute for the specified LUN	7, C
Get-N?LunSerialNumber	Gets the serial number for the specified LUN	7, C
Get-N?LunSignature	Gets the MBR or GPT signature of a partitioned LUN	7, C
Get-NaLunSnapUsage	Lists all LUNs backed by data in the specified Snapshot copy	7
Get-NaLunSpaceReserved	Queries the space reservation settings for the named LUN	7
Get-N?LunStatistics	Gets block protocol access statistics, in bytes, for LUNs	7, C
Get-NaLunTargetDeviceId	Gets the SCSI target device ID	7
New-N?Lun	Creates a LUN by one of three methods: by size, from file, or from Snapshot copy	7, C
New-NaLunClone	Creates a LUN clone	7
New-NcLunImportRelationship	Creates a LUN import relationship with the purpose of importing foreign disk data into the LUN	C
Remove-N?Lun	Destroys the specified LUN	7, C
Remove-NcLunImportRelationship	Deletes the import relationship of the specified LUN or the specified foreign disk	C
Remove-N?LunMap	Reverses the effect of <code>Add-NaLunMap</code> on the specified LUN for the specified group	7, C
Remove-NcLunMapReportingNodes	Removes nodes from the list reporting the LUN map	C
Rename-N?Lun	Moves (renames) a LUN	7, C
Resume-NcLunCopy	Resumes a paused LUN copy operation	C
Resume-NcLunImport	Resumes a paused LUN import operation	C
Resume-NcLunMove	Resumes a paused LUN move operation	C
Set-N?Lun	Sets the specified LUN to online or offline	7, C
Set-N?LunAttribute	Sets a named attribute for a given LUN	7, C
Set-N?LunComment	Sets the optional descriptive comment for a LUN	7, C
Set-NcLunCopyMaxThroughput	Modifies the maximum throughput of an ongoing copy operation	C
Set-NaLunDeviceId	Sets or clears the SCSI device identifier for the LUN	7
Set-NcLunImportThrottle	Modifies the max throughput limit for the specified import relationship	C

Command	Function	Available In
Set-NcLunMoveMaxThroughput	Modifies the maximum throughput of an ongoing move operation	C
Set-NcLunQosPolicyGroup	Sets the QoS policy group for a LUN	C
Set-N?LunSelect	Sets the select attribute for the specified LUN	7, C
Set-N?LunSerialNumber	Sets the serial number for the specified LUN	7, C
Set-NaLunShare	Enables file system protocol-based access to a LUN	7
Set-N?LunSignature	Sets the MBR or GPT signature of a partitioned LUN	7, C
Set-N?LunSize	Changes the size of the LUN	7, C
Set-NcLunSpaceAllocated	Sets the space allocation settings for the named LUN	C
Set-N?LunSpaceReserved	Sets the space reservation settings for the named LUN	7, C
Start-NaLunCloneSplit	Starts the cloning of the given LUN	C
Start-NcLunCopy	Starts copying a group of LUNs from one volume to another	C
Start-NcLunImport	Performs a start-and-import operation for the specified LUN	C
Start-NcLunImportVerify	Starts the verification for the specified LUN	C
Start-NcLunMove	Starts a LUN move operation to move one or more LUNs from one volume to another within an SVM	C
Stop-NcLunCopy	Cancels an ongoing LUN copy operation before creation of the new LUN	C
Stop-NaLunCloneSplit	Stops the cloning of the given LUN	7
Stop-NcLunImport	Aborts an import operation for a specified LUN	C
Stop-NcLunImportVerify	Stops the verification for a specified LUN	C
Stop-NcLunMove	Cancels an ongoing LUN move operation before creation of the new LUN	C
Suspend-NcLunCopy	Pauses an ongoing LUN copy operation	C
Suspend-NcLunImport	Pauses an ongoing LUN import operation	C
Suspend-NcLunMove	Pauses an ongoing LUN move operation	C
Test-NaLunConfig	Lists configuration warnings for various use cases	7

4.5 Specific Volume Commands

Understanding how to manage volumes is an integral part of managing a NetApp controller or SVM. Almost all of the policy-based controls on the storage are assigned from the volume and are inherited by the child object, which are LUNs and shares/exports. These policies include Snapshot copy schedules, deduplication settings, thin-provisioning settings and space guarantees, and defining the location of the

storage regarding the aggregate on which a LUN is housed. ONTAP adds a major feature called vol move. Vol move allows you to nondisruptively relocate a volume and all of its LUNs and shares/exports from a current set of media to a different set of media. You can do this relocation on either the same controller or another controller in the cluster.

1. Interrogate the array to get a list of all of the volumes.

```
PS:> Get-NcVol
```

2. If you have free space on an aggregate and know the volume name that you will use, you can create a new volume in the SVM into which you are currently logged. In the following example, the volume name is voltemp, the aggregate is aggr1_node01, and the size is 100GB.

```
PS:> New-NcVol voltemp aggr1_node01 100g /voltemp #C-Mode version
PS:> New-NaVol voltemp aggr1_node01 100g #7-Mode version
```

3. In the previous example, the commands were given in order. The following example is the expanded version of the same command:

```
PS:> New-NcVol -Name voltemp -Aggregate aggr1_node01 -size 100g -junctionpath /voltemp
```

4. Deleting a volume requires that the volume be empty and offline. Run the LUN, Share and Export commands to remove volume contents. Then run the following series of commands to delete a volume:

```
PS:> Set-NcVol Volume31 -offline # C-Mode version
PS:> Remove-NcVol Volume31 # C-Mode version
PS:> Set-NaVol Volume31 -offline # 7-Mode version
PS:> Remove-NaVol Volume31 # 7-Mode version
```

5. Modify the size of the volume.

```
PS:> Set-NcVolSize Volume31 +100g
```

6. To change the individual settings on a volume, such as increasing space, run the following commands. Run the Get command to gather a list of valid changeable options.

```
PS:> Get-NcVolOption volume31
```

7. Run the Set command to set those options to new values.

```
PS:> Set-NcVolOption volume31 nosnapdir off
PS:> Set-NcVolOption volume31 guarantee none
```

Some common volume command options include:

- Fractional_reserve (a number from 0 to 100)
- Nosnapdir (set to on or off)
- Guarantee (set to none, file, or volume)

Table 9 shows a complete list of volume commands. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 9) Complete list of volume commands.

Command	Function	Available In
Dismount-NcVol	Unmounts a volume from its parent volume and deletes the junction path at which the volume is mounted	C
Get-N?Vol	Gets the volume status	7, C
Get-NcVolAutobalance	Displays information about autobalance volume operations for an Infinite Volume	C

Command	Function	Available In
Get-N?VolAutosize	Gives the name of a flexible volume; gets the autosize settings	7, C
Get-NaVolCharmap	Returns charmap information for a specified volume	7
Get-NcVolClone	Displays a list of NetApp FlexClone® volumes	C
Get-N?VolCloneSplit	Displays the progress in separating clones from their underlying parent volumes and Snapshot copies	7, C
Get-N?VolCloneSplitEstimate	Displays an estimate of additional storage in the underlying aggregate required to perform a volume clone split operation	7, C
Get-N?VolContainer	Returns the name of the containing aggregate for the named flexible volume	7, C
Get-N?VolFootprint	Gets a list of volumes and a breakdown of their data and metadata footprints in their parent aggregates	7, C
Get-N?VolLanguage	Gets language mapping for the given volume	7, C
Get-NcVolLimit	Returns calculated volume limits that are based on the current configuration of the SVM	C
Get-N?VolMove	Obtains the status of a volume move; if a volume name is not specified, all in-progress volume move operations are returned	7, C
Get-NcVolMoveTargetAggr	Scans aggregates and returns a list of compatible target aggregates for the given volume move operation	C
Get-N?VolOption	Gets the options that have been set for the specified volume	7, C
Get-N?VolRoot	Gets the root volume for the current controller	7, C
Get-NcVolSize	Gets the size of a volume	C
Get-NaVolSis	Gets the dedupe status of an SIS volume	7
Get-N?VolSize	Gives the name of a flexible volume and returns its current size, in bytes	7, C
Get-N?VolSpace	Gets a list of volumes and a breakdown of their space usage	7, C
Get-NcVolStorageService	Gets a list of volume storage service objects	C
Invoke-N?VolMoveCutover	Initiates a manual cutover operation; if a cutover cannot be initiated or completed, the API returns with an error	7, C
Mount-NcVol	Mounts a volume on another volume (parent) with a junction path	C
New-N?Vol	Creates a new flexible volume with the given name and characteristics	7, C
New-N?VolClone	Creates a flexible volume that is a clone of a backing or a parent flexible volume	7, C
Remove-N?Vol	Destroys the specified volume or plex	7, C
Rename-N?Vol	Renames the specified volume	7, C
Resume-NaVolMove	Resumes a previously paused move	7

Command	Function	Available In
Set-N?Vol	Sets the specified volume to online, offline, or restricted	7, C
Set-N?VolAutosize	Gives the name of a flexible volume; sets the autosize settings	7, C
Set-NaVolCharmap	Associates a charmap description with a specified volume	7
Set-NaVolLanguage	Sets the language mapping for the given volume	7
Set-N?VolOption	Sets the key option to the value specified by value in the specified volume	7, C
Set-N?VolSize	Gives the name of a flexible volume; sets the size of the volume to the stated amount	7, C
Set-N?VolTotalFiles	Sets a total files value of the volume to the given quantity	7, C
Start-NcVolAggrVacate	Moves all constituents belonging to a given Infinite Volume from the source aggregate to the destination aggregate	C
Start-NcVolAutobalance	Starts an autobalance volume operation for an Infinite Volume	C
Start-N?VolCloneSplit	Starts the process by which the given clone is split from its underlying parent volume and Snapshot copies	7, C
Start-N?VolMove	Moves a single 7-Mode FlexVol volume between two aggregates on the same controller	7, C
Stop-NcVolAutobalance	Allows the user to stop an autobalance volume operation on an Infinite Volume	C
Stop-N?VolCloneSplit	Stops the process of splitting a clone from its parent volume and Snapshot copies	7, C
Stop-N?VolMove	Stops the volume move operation	7, C
Suspend-NaVolMove	Suspends the volume move operation	7
Update-NcVol	Modifies the attributes of a volume or a group of volume objects	C

4.6 Host-Based LUN Interrogation Commands

After the LUN is assigned to the server, complete the following steps on the server to make the disk usable:

1. If it is offline, bring the disk online.
2. If it is not already initialized, initialize the disk.

Note: Part of the initialization process is to determine the disk type (basic or dynamic) and the format style (MBR or GPT).

3. Partition the drive.

Note: The partitions can be formatted with the NTFS or ReFS file systems.

Each of these steps, and how to accomplish them using the PSTK, is described in the following PowerShell command sets.

LUN Configuration Commands

Connect the NetApp LUN to the Windows Server by running the `Get-NcHostDisk` command. This command detects 7-Mode and clustered Data ONTAP LUNs and identifies the controller (7-Mode) or SVM (clustered Data ONTAP) as well as the path to the LUN.

```
PS:> Get-NcHostDisk
HostDrivePath  Disk  Size           ControllerPath
C:\            0    136.2GB
D:\            1    100GB          Worker-A:/vol/vol3/DataLUN1
E:\            2    130GB          Infra:/vol/ExtraVMs/TestVM1
C:\MountPoint1 4    143GB          Infra:/vol/Logs/LogSrv1
               5    300GB          Shift:/vol/Templates/VMDKSource
```

In this example:

- The `C:\` drive is not hosted on a NetApp controller and is referred to by Windows in the disk management applet as `WinDisk 0`.
- `WinDisk 1` is mounted as drive `D:\` and is hosted on a controller named `Worker-A:` in the `vol3` volume. The LUN name is `DataLUN1`.
- The mount points and drive letters were discovered. The `WinDisk` numbers might contain gaps, which indicate that a drive that was previously mapped to `WinDisk 3` could have been unmapped from the host.
- A newly mapped LUN mounted as `WinDisk 5` (no drive letter or mount point).

If a newly mapped drive is not yet visible, you can force a Windows plug-and-play (PnP) rescan by running the following command.

Note: PnP-type scans can take up to 90 seconds to discover newly mapped LUNs. You can force the rescan to wait until the drive shows up to allow scripts to operate correctly when later steps depend on a specific LUN.

```
PS:> Start-NcHostDiskRescan # Kick off a manual rescan.
PS:> Wait-NcHostDisk -ControllerLunPath /vol/Vol3/Lun5 -Controllername infra
```

A common use of the `wait` command is for a timeout to wait for a specific LUN to appear. For example, run the `-timeout xxx` command, where `xxx` is the wait period in milliseconds. A timeout value of 90,000 would be 90 seconds.

Another common command is the settling time, which is the specified amount of time when a disk must remain unchanged before it is considered stable and usable. In the case of multipathing MPIO environments, this command is valuable. That is because when a disk with all of its paths appears, this disk appears as unique entries until the MPIO driver removes the extra paths from the `WinDisk` list. Running the settling time command can prevent this. A common settling time is 10,000, which represents 10 seconds.

In addition, a host can use the LUN serial number instead of the controller name and path as the factor to determine if a drive is detected on a host. This command is called `-SerialNumber`.

1. After the drive is discovered, pipeline it to the next command that initializes the drive, as shown in the following example. This example also shows the command to initialize all detected uninitialized disks.

```
PS:> Initialize-NcHostDisk -DiskIndex 4 -partitionstyle GPT
PS:> Get-NcHostDisk -uninitialized | Initialize-NcHostDisk -partitionstyle GPT
```

2. To initialize a specific LUN when you don't know the `WinDisk` number, discover the element by using the LUN serial number and pipeline.

```
PS:> Get-NcHostDisk | where-object{$_.DiskSerialNumber -contains "HnT7cJYxOeIX"} | initialize-
NcHostDisk -partitionstyle gpt
```

3. After the drive is initialized, install a new partition on the drive and format that drive with a file system.

```
PS:> New-NcHostVolume -diskindex 4 -label MyBigDrive -mount X
```

Note: By default, if no size is specified by using the `-Size` command, then the entire LUN is used for the host volume. If no `-Mount` is used, then the next available drive letter is used.

4. If you don't know the WinDisk number or you want to force a specific drive letter to a specific LUN, use a pipelined object similar to the one used for initializing the drive to force the behavior.

```
PS:>Get-NcHostDisk | where-object{$_ .DiskSerialNumber -contains "HnT7cJYxOeIX"} | New-NcHostVolume -Mount X
```

LUN Maintenance and Operational Commands

LUN maintenance and operational commands alter the way that a LUN is maintained over time. These commands fall under the classification of sub-LUN clone commands and space reclamation commands, which can be used to retain free space.

If, for example, you deploy a 1TB LUN to a host and that host is constantly writing and overwriting that LUN, then, over time, the thin space on that LUN is slowly consumed. This is because the OS of the host does not inform the storage device.

For example, as a LUN ages, the space allocated to the LUN by the NetApp controller can greatly exceed the space used by the file system on the LUN because of the host OS not informing the storage device which blocks are no longer needed. Using PSTK, you can run the following command, which identifies all of the unused blocks in the file system and sends that list to the storage device. The storage device can then unmap these blocks and return the space to the thin pool.

```
PS:> Invoke-NcHostVolumeSpaceReclaim X:\
```

A lighter-weight version of this command can perform the same type of operation for individual files. You might have a file system with a collection of very large files (for example, VHDs) and you want to delete one of these large files. By running the `Remove-NcHostFile` command, PSTK gathers the list of blocks that the file consumes on the array and only unmaps those individual blocks. Run the following command to accomplish this task:

```
PS:> Remove-NcHostFile X:\Directory4\File3.vhdx
```

In addition to commands that recover unused blocks, there is a command that makes a copy of a file that takes no space on the array. If you have a very large file and want to copy it to another location with a new file name, you can run the `Copy-NcHostFile` command. Doing so creates a clone of the blocks that represent that file on the LUN. This presents the new writable file almost immediately while consuming almost no space on the LUN. This command can clone the file from any location on a LUN to another location on the same LUN or from any location on a LUN to any location on a different LUN that exists in the same NetApp FlexVol volume.

```
PS:> Copy-NcHostFile X:\DirectoryX\File1.VHDx X:\DirectoryZ\File6.VHDx
```

The following example shows where the commands work and where they do not work:

```
PS:> Get-NcHostDisk
HostDrivePath  Disk  Size           ControllerPath
-----
C:\            0     136.2GB
E:\            2     130GB          Infra:/vol/ExtraVMs/TestVM1
C:\MountPoint1 4     143GB          Infra:/vol/Logs/LogSrv1
Z:\            4     400GB          Infra:/vol/Logs/TempSrv4

PS:> Copy-NcHostDisk Z:\File1.vhd Z:\Directory1\File5.VHD
# This command will work as expected

PS:> Copy-NcHostDisk Z:\File1.vhd C:\MountPoint1\Directory1\File5.VHD
# This command will work as expected, exist in same FlexVol named 'Logs'
```

```
PS:> Copy-NcHostDisk C:\MountPoint1\File23.VHD E:\Test.VHD
# This command won't work, since they don't exist on the same FlexVol
```

Creating and Maintaining Virtual Disks

Creating a fixed virtual hard disk (VHD) or fixed virtual hard disk format (VHDX) inside of Windows Server requires zeroing the disks. This zeroing process creates a large amount of useless I/O activity and consumes storage device resources for no appreciable value. To speed up this process, avoid the zeroing process, and still protect the ability to present a zeroed block, run the following commands:

```
PS:> Net-NcVirtualDisk X:\Folder1\File1.vhd 1tb
PS:> Net-NcVirtualDisk X:\Folder1\File1.vhd 1tb -vhdx
```

The process to create this VHD and VHDX decreases the time required from hours to seconds while consuming less than 1% of the storage resources that would otherwise be used.

An advantage of using dynamic VHDs and VHDXs is that they can be created instantly and can grow and shrink over time. A disadvantage is that dynamic VHDs offer lower performance than the alternative fixed VHDs. The previous command shows how fixed VHDs and VHDXs can be created immediately. The following command shows how to quickly grow or shrink a fixed VHD:

```
PS:> Set-NcVirtualDiskSize X:\VM1.Vhdx +100g # Grows VHD (and Partition inside it) by 100GB
PS:> Set-NcVirtualDiskSize X:\VM1.Vhdx -35g # Decrease VHD (and Partition) by 35GB
PS:> Set-NcVirtualDiskSize X:\VM1.Vhdx +10% # Increase VHD (and Partition) by 10%
PS:> Set-NcVirtualDiskSize X:\VM1.Vhdx -minimum # Decrease VHD (and Partition) to smallest size
```

Note: A partition can be shrunk only to the smallest amount of contiguous space that has been written to.

To combine common VHD commands with space reclamation, run the `Invoke-NcHostVolumeSpaceReclaim` command.

The following command enables you to reclaim free space inside a VHD:

```
PS:> Mount-DiskImage -imagepath "X:\FileVHD.vhd"
```

To get the drive letter from the disk, pipe the preceding command. This command can then be pipelined directly to the space reclamation command.

```
PS:> Mount-DiskImage -imagepath "X:\FileVHD.vhd" | get-disk | get-partition | get-volume |
Invoke-NcHostVolumeSpaceReclaim
```

4.7 Aggregate and Disk Commands

You can run aggregate commands against 7-Mode and clustered Data ONTAP controllers, but they are not accessible for SVMs. The most common commands used to manage aggregates are based on creating aggregates, modifying aggregate settings or capacity, removing aggregates, and validating spares and parity.

To create a new aggregate, run the `New-NcAggr` command.

Note: Most of the settings are filled in using the controller defaults unless otherwise specified.

```
PS:> New-NcAggr Aggr1_Loki01 -node Loki_Node1 -diskcount 7
PS:> New-NaAggr Aggr1 -diskcount 7
```

These commands are based on the following assumptions:

- The first seven drives of the same disk type and RPM are selected.
- The RAID type defaults to RAID_DP.

- The RAID group size for each plex defaults to the correct value, such as 20 for SAS drives.
- The RAID block type defaults to 64-bit aggregates instead of 32-bit.

You can override the default behaviors by specifying the exact disks you want to add or by specifying overrides for each of these values. Examples of these overrides are listed in Table 10.

Table 10) Default overrides.

Options			
-DiskSize	-DiskType	-Raidsize	-RPM
-AllowMixedRPM	-RaidType	-Blocktype	-ForceSparePool

After an aggregate is created, you can add more drives to the RAID group. By default, the `Add` command attempts to completely fill out existing plexes before creating additional plexes. Adding disks to an existing aggregate does not affect performance or require any restriping process.

```
PS:> Add-NcAggr Aggr1 -diskcount 4
```

Table 11 shows the complete list of aggregate commands. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 11) Complete list of aggregate commands.

Command	Function	Available In
Add-N?Aggr	Adds disks to the specified aggregate	7, C
Confirm-N?AggrSpareLow	Returns true if there is no suitable spare disk available for any file system (parity or data) disk	7, C
Disable-NcAggrAutobalance	Disables aggregate autobalance	C
Enable-NcAggrAutobalance	Enables aggregate autobalance	C
Get-N?Aggr	Gets the aggregates for the current controller	7, C
Get-NcAggrAutobalance	Gets the configuration for the autobalance aggregate	C
Get-NcAggrAutobalanceAggrState	Gets a list of aggr autobalance objects	C
Get-NcAggrAutobalanceNotification	Gets the nodes that notify the autobalance aggregate about aggregates that need to be balanced	C
Get-NcAggrAutobalanceVolumeState	Gets the aggregate autobalance volume state	C
Get-NaAggrFilerInfo	Gets information about what possibilities and parameters exist for aggregates on a given storage system	7
Get-NcAggrNodeInfo	Gets information about what possibilities and parameters exist for aggregates on a given controller	C
Get-NaAggrMediaScrub	Gets the status of media scrubbing on the named aggregate, plex, or RAID group	7
Get-N?AggrOption	Gets the options that have been set for the specified aggregate	7, C

Command	Function	Available In
Get-NcAggrRelocation	Gets the aggregate relocation status	C
Get-N?AggrScrub	Gets the status of parity scrubbing on the named aggregate, plex, or RAID group	7, C
Get-N?AggrSpace	Shows the space usage of the aggregate on a per-flexible volume basis	7, C
Get-NcAggrSpare	Gets a list of spare disks	C
Get-N?AggrVerify	Gets the status of RAID mirror verification on the named aggregate	7, C
Get-NcAggrStatus	Gets status and topology information about one or more aggregates from the RAID subsystem	C
Move-NcAggr	Gets status and topology information about one or more aggregates from the RAID subsystem	C
New-N?Aggr	Creates a new aggregate	7, C
New-NaAggrMirror	Turns an unmirrored aggregate into a mirrored aggregate by adding a plex to it	7
Remove-N?Aggr	Destroys the specified aggregate or plex	7, C
Rename-N?Aggr	Renames the specified aggregate	7, C
Resume-N?AggrScrub	Resumes parity scrubbing on the named aggregate, plex, or RAID group	7, C
Resume-NaAggrVerify	Resumes RAID mirror verification on the named aggregate	7
Set-N?Aggr	Sets the specified aggregate or plex status to online, offline, or restricted	7, C
Set-N?AggrOption	Sets a specified option for the given aggregate	7, C
Set-N?AggrRaidType	Gets status and topology information about one or more aggregates from the RAID subsystem	7, C
Split-N?AggrMirror	Removes the specified plex from a mirrored aggregate and creates a new unmirrored aggregate with the specified name that contains the plex	7, C
Start-N?AggrScrub	Starts parity scrubbing on the named aggregate, plex, or RAID group	7, C
Start-NaAggrVerify	Starts the RAID mirror verification on the named aggregate	7
Stop-N?AggrScrub	Stops parity scrubbing on the named aggregate, plex, or RAID group	7, C
Stop-NaAggrVerify	Stops the RAID mirror verification on the named aggregate	7
Suspend-N?AggrScrub	Suspends parity scrubbing on the named aggregate, plex, or RAID group	7, C

Command	Function	Available In
Suspend-NaAggrVerify	Suspends the RAID mirror verification on the named aggregate	7

Disk Commands

If the drive being added was previously zeroed, the process should only take minutes. However, if the drives need to be zeroed, this process can take many hours. To force the controller to go through the zeroing process on a non-zeroed drive, run the following command.

Note: By default, when a new controller is deployed, it automatically zeroes its drives. The common way to encounter non-zeroed drives is by creating an aggregate and then destroying that aggregate later, which allows the drive to reenter the spare pool as a non-zeroed spare drive.

```
PS:> Start-NcDiskZeroSpare
```

An array zeroes its spares; it also updates firmware on drives when a new controller is installed or updated to a new version of ONTAP. To force a drive firmware update at any time, run the following command:

```
PS:> Start-NcDiskUpdate
```

Another common practice is to decommission drives from a cluster or to reown the drives from one node of a cluster to its partner node in that cluster.

```
PS:> Clear-NcDiskOwner Loki_Node-01:01.23.14 # Remove ownership, ie decommission
PS:> Get-NcNode Loki_Node-01 | get-NcDiskOwner -OwnershipType unowned | Set-NcDiskOwner
```

Table 12 shows the complete list of disk commands. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 12) Complete list of disk commands.

Command	Function	Available In
Clear-NcDiskForeign	Marks a disk as not foreign	C
Clear-N?DiskOwner	Removes ownership information about a disk	7, C
Get-N?Disk	Gets disk information about one or more disks	7, C
Get-N?DiskOwner	Gets disk ownership information	7, C
Get-N?DiskOwnerFiler	Gets storage system information	7, C
Remove-N?Disk	Removes a spare disk	7, C
Set-N?DiskFail	Fails a file system disk	7, C
Set-NcDiskForeign	Marks an array LUN as a foreign disk	C
Set-N?DiskOwner	Changes ownership on a disk	7, C
Set-N?DiskUnfail	Unfails a disk in the broken pool	7, C

Command	Function	Available In
Start-NaDiskUpdate	Starts the disk firmware download process to update firmware on disks	C
Start-N?DiskZeroSpare	Sets up all nonzeroed spares owned by the storage system to start zeroing	7, C
Stop-NaDiskReplace	Aborts disk replace	7
Switch-N?DiskOwner	Changes ownership on disks already belonging to an owner	7, C

4.8 Common SMB Commands

There are differences between the 7-Mode and clustered Data ONTAP implementations of the SMB protocol. Although the 7-Mode controller supports SMB 1, SMB 2, and SMB 2.1, clustered Data ONTAP supports all of these modes as well as SMB 3.0. The following commands are used to manage and modify the CIFS service on the SVM.

Note: When deploying a CIFS server on clustered Data ONTAP, these commands must be run against the SVM and not the cluster, because these are all SVM settings.

Note: By default, the following commands should be run in the following sequence to start an SMB share on the NetApp controller (either a 7-Mode controller or a clustered Data ONTAP SVM).

```
PS:> Add-NcCifsServer -Name Cifs01 -Domain TtestDOMAIN -AdminUsername Administrator -
AdminPassword p@ssword # command used for CDOT to set and join AD

PS:> Set-NaCifs -CifsServer Cifs01 -AuthType ad -SecurityStyle ntfs -Domain TestDOMAIN -User
Administrator -Password p@ssword # command used to set the AD
PS:> Enable-NaCifs # Extra step required to start the Cifs Service on the 7-Mode controller
```

Use the commands shown in Table 13 to create and join your CIFS service to the active domain.

Table 13) Commands for joining CIFS service to active domain.

7-Mode Command	Function	Clustered Data ONTAP Command
Enable-NaCifs	Starts the CIFS service	Add-NcCifsServer Start-NcCifsServer
Get-NaCifs	Displays the CIFS configuration	Get-NcCifsServer
Set-NaCifs	Configures the CIFS services	Set-NcCifsServer
Disable-NaCifs	Stops the CIFS service for all users or a particular workstation, if specified	Stop-NcCifsServer
Test-NaCifs	Returns running state of CIFS service	
	Removes a CIFS server	Remove-NcCifsServer

In general, the default options for an SMB share are appropriate for most situations. However, some SMB options are commonly changed depending on the customer's site requirements.

Examples of these common commands are shown in Table 14. In this table, C represents clustered Data ONTAP.

Table 14) Example commands for configuring SMB shares.

Command	Function	Available In
Get-NcCifsOption	Gets the CIFS-specific tunables that can be set on an SVM	C
Get-NcCifsSecurity	Gets the CIFS security tunable parameters	C
Set-NcCifsOption	Modifies the CIFS-specific tunables that can be set on an SVM	C
Set-NcCifsSecurity	Sets the CIFS security tunable parameters	C

To set the options on the `Set-NcCifsOptions` command, run the command this way:

```
PS:> Set-NcCifsOption -optionname optionvalue
```

Valid option names are shown in Table 15. Run the following command to return all of the valid option names. The option names that follow are only the most commonly used names.

```
PS:> get-help Set-NcCifsOption -detailed
```

Table 15) Valid option names.

Option Names			
DefaultUnixUser	isSMB3Enabled	IsCopyOffloadEnabled	IsDACEEnabled
IsAdvancedSparseFileSupportEnabled	MaxFileWriteZeroLength	vServerContext	

The natural follow-on to configuring the SMB service is to set up a connection either to expose the functionality to a set of Active Directory users (7-Mode or clustered Data ONTAP) or to define access using local-only type accounts in workgroup mode (only available on 7-Mode). Table 16 lists the commands for domain management and NetBIOS names. They are optional and are used only to tune a CIFS service installation by defining which domain or DNS server a CIFS service should use first or to help troubleshoot problems joining specific AD environments. In Table 16, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 16) Commands for domain management and returning NetBIOS names.

Command	Function	Available In
Add-NcCifsPreferredDomainController	Adds to the list of preferred domain controllers	C
Add-NcCifsTrustedDomain	Adds to the list of trusted domains for name mapping search.	C
Find-NcCifsDomainServer	Triggers the discovery of domain servers	C
Get-NcCifsDomainServer	Gets the list of servers discovered by the cluster	C
Get-NcCifsDomainTrust	Retrieves the list of discovered trusted domains	C

Command	Function	Available In
Get-NaCifsNetBiosAlias	Returns the current list of NetBIOS alias names for the controller	7
Get-NcCifsNbtStat	Gets the NetBIOS name service stats for SVMs in cluster	C
Get-NaCifsOrganizationalUnit	Gathers a list of joinable organizational units from AD	7
Get-NcCifsPreferredDomainController	Gets list of preferred domain controllers in an AD domain	C
Get-NaCifsSite	Gathers the list of joinable site units from AD	7
Get-NcCifsTrustedDomain	Retrieves the list of trusted domains for name-mapping search	C
Invoke-NcCifsDomainTrustsDiscovery	Triggers rediscovery of trusted domains	C
Remove-NcCifsPreferredDomainController	Removes from a list of preferred domain controllers	C
Remove-NcCifsTrustedDomain	Removes from the list of trusted domains for name-mapping search	C
Set-NaCifsNetBiosAlias	Provides a list of NetBIOS alias names for the controller	7
Set-NcCifsTrustedDomain	Modifies the list of trusted domains for name-mapping search	C
Test-NaCifsName	Detects if CIFS server name is already used in network/domain	7

The CIFS service itself doesn't actually create the individual shares. To create individual SMB shares, run the following command:

```
PS:> Add-NcCifsShare -Name Share1 -Path /CifsVoll # similar for 7/C
```

Table 17 lists the common commands for CIFS share management. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 17) Commands for CIFS share management.

Command	Function	Available In
Add-N?CifsShare	Creates a new CIFS share rooted at the specified path	7, C
Get-N?CifsShare	Gets a list of CIFS shares	7, C
Remove-N?CifsShare	Removes the specified CIFS share	7, C
Set-N?CifsShare	Modifies settings of a CIFS share, even if the share is in use	7, C

Table 18 list the commands for setting and modifying passwords. These commands exist to support sites that enforce password change requirements and allow the NetApp CIFS service to work in those

environments. In Table 18, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 18) Commands for setting and modifying passwords.

Command	Function	Available In
Disable-NcCifsDomainPasswordSchedule	Disables the CIFS domain password schedule	C
Enable-NcCifsDomainPasswordSchedule	Enables the CIFS domain password schedule	C
Get-NcCifsDomainPasswordSchedule	Gets a list of CIFS domain password schedule objects	C
New-NcCifsPassword	Generates a new password for the CIFS server's machine account and changes it in the Windows Active Directory domain	C
New-NaCifsPasswordFile	Creates a basic <code>/etc/passwd</code> file, including a root user with the specified password	7
Reset-NcCifsPassword	Resets the CIFS server's machine account password in the Windows Active Directory domain	C
Set-NcCifsDomainPasswordSchedule	Modifies the attributes of the CIFS domain password schedule object	C
Test-NaCifsPasswordGroupFile	Determines whether the <code>/etc/passwd</code> and <code>/etc/group</code> files, which would be used in the event that the configured CIFS authentication method fails, exist	7

Table 19 lists the commands for setting up and using access control lists (ACLs) and privileges to control access. To grant permissions to a specific user or group on a share, run the following commands:

```
PS:> Add-NcCIFSACL -Share CifsVoll -UserOrGroup Users -Permission read
PS:> Add-NcCifsPrivilege -Share CifsVoll -UserOrGroup Users -Privilege SetTakeOwnership
```

Note: Valid permission types are `No_Access`, `Read`, `Change` and `Full_Control`.

Note: Valid privileges are `SetCBPrivilege`, `SetBackupPrivilege`, `SetRestorPrivilege`, `SetTakeOwnership`, and `SetSecurityPrivilege`.

In Table 19, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 19) Commands for setting up and using ACLs and privileges to control access.

Command	Function	Available In
Add-NcCifsPrivilege	Adds privileges to a local or an Active Directory user or group	C
Add-NcCifsShareAcl	Adds permissions for a user or group for a defined CIFS share	C
Get-NcCifsPrivilege	Retrieves the list of local groups	C

Command	Function	Available In
Get-N?CifsShareAcl	Gets a list of permissions on defined CIFS shares	7, C
Remove-NcCifsPrivilege	Removes privileges from a local or an Active Directory user or group	C
Remove-N?CifsShareAcl	Removes permissions for a user or group on a defined CIFS share	7, C
Reset-NcCifsPrivilege	Resets privileges for a local or an Active Directory user or group	C
Set-N?CifsShareAcl	Sets the permissions for a user or group on a defined CIFS share	7, C

Table 20 lists the commands for setting permissions for users and groups on specific shares. Users and groups come in two types: locally defined users and groups and domain-defined users and groups.

```
PS:> New-NcCifsLocalGroup -Name "PowerShell Users" -Description "My Friends"
PS:> New-NcCifsLocalUser -Name "Chris" -FullName "Chris Lionetti" -vServer MySVM
PS:> Add-NcCifsLocalGroupMember -Name "PowerShell Users" -Member "Chris"
PS:> Set-NcCifsLocalUser -Name "Chris" -Password (Convertto-SecureString "p@ssword" -force -
asplaintext)
```

Note: This set of commands uses a special method for inserting the password into the script. By default, the password option for the `Set-NcCifsLocalUser` command requires a type of `SecureString`, which means it won't accept simple text without forcing it into the `SecureString` type. If the same command is run without the password option, the command prompts for the password. This method illustrates how, when creating many users, having a pop-up for each user is hard to manage.

In Table 20, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 20) Commands for setting permissions for users and groups on specific shares.

Command	Function	Available In
Add-NcCifsLocalGroupMember	Adds local users or Active Directory users/groups to a local group	C
Get-NcCifsLocalGroup	Gets a list of CIFS local groups	C
Get-NcCifsLocalGroupMember	Gets the list of local group members	C
Get-NcCifsLocalUser	Gets the list of CIFS local users	C
New-NaCifsGroupFile	Creates a basic <code>/etc/group</code> file	7
New-NcCifsLocalGroup	Creates a CIFS local group	C
New-NcCifsLocalUser	Configures and creates a CIFS local user	C
Remove-NcCifsLocalGroup	Removes a CIFS local group	C
Remove-NcCifsLocalGroupMember	Removes local users or Active Directory users/groups from a local group	C

Command	Function	Available In
Remove-NcCifsLocalUser	Deletes a CIFS local user	C
Rename-NcCifsLocalGroup	Changes the name of a CIFS local group	C
Rename-NcCifsLocalUser	Renames a CIFS local user	C
Set-NcCifsLocalGroup	Modifies a CIFS local group	C
Set-NcCifsLocalUser	Modifies a CIFS local user	C

After an SMB share is used by clients, each active connection is considered a session and these sessions can be managed granularly. Table 21 lists the CIFS sessions and statistics of a running SMB share. These commands are only needed to force specific behavior on the SVM or 7-Mode controller; use them sparingly because they can disconnect active hosts. In Table 21, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 21) Commands for CIFS sessions and statistics.

Command	Function	Available In
Close-NcCifsSession	Closes an open CIFS session	C
Close-NcCifsSessionFile	Closes an open CIFS file	C
Get-N?CifsSession	Retrieves the list of the established CIFS sessions	7, C
Get-NcCifsSessionFile	Retrieves the list of the opened CIFS files	C
Get-NaCifsStatistics	Displays CIFS client statistics	7

The capability to create a VSS-based Snapshot copy is a new feature of SMB 3. This feature is available only as a clustered Data ONTAP option. Table 22 lists the commands for VSS-based Snapshot copy management. In this table, 7 stands for Data ONTAP operating in 7-Mode and C stands for clustered Data ONTAP.

Table 22) Commands for VSS-based Snapshot copy management.

Command	Function	Available In
Add-NcCifsShadowCopyFile	Adds a list of files to shadow copy in a particular share	C
Get-NcCifsShadowCopyEmsMessage	Gets shadow copy EMS messages	C
Restore-NcCifsShadowCopyDirectory	Requests that the storage system restore a directory	C
Save-NcCifsShadowCopySnapshot	Requests that the storage system keep the Snapshot copies taken as part of the shadow copy set creation	C

Table 23 lists the commands for home directory management. For more information about best practices for the setup and configuration of home directories on a NetApp controller, see [“NetApp TR-3771: Windows File Services Best Practices with NetApp Storage Systems.”](#)

In Table 23, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 23) Commands for home directory management.

Command	Function	Available In
Add-NcCifsHomeDirectorySearchPath	Adds a path to list of paths searched to find a user's home directory	C
Get-NcCifsHomeDirectoryConfig	Gets the CIFS home directory configurations	C
Get-NcCifsHomeDirectorySearchPath	Gets a list of CIFS home directory search paths	C
Get-NcCifsUserHomeDirectory	Gets a user's home directory based on user name	C
Remove-NcCifsHomeDirectorySearchPath	Removes a home directory search path	C
Set-NaCifsHomeDirectory	Provides a list of CIFS home directory paths for the controller	7
Set-NcCifsHomeDirectoryConfig	Modifies CIFS home directory configurations	C
Set-NcCifsHomeDirectorySearchPath	Sets the position of a path in the list of paths that is searched to find a CIFS user's home directory	C

Table 24 lists the commands for BranchCache services management.

For more information about how to set up and configure BranchCache, see the [Microsoft TechNet](#) site.

For more information about supporting BranchCache, see the [NetApp Support](#) site.

In Table 24, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 24) Commands for BranchCache services management.

Command	Function	Available In
Get-NcCifsBranchCache	Gets the CIFS BranchCache configurations	C
Get-NaCifsBranchCacheHashStatistics	Gets the CIFS BranchCache statistics	7
Invoke-NcCifsBranchCacheFlush	Flushes (deletes) all of the BranchCache hashes that were generated for an SVM	C
Invoke-NcCifsBranchCacheHash	Forces the generation of BranchCache hashes for a file or path	C
New-NcCifsBranchCache	Creates and sets up the BranchCache service	C
Remove-NcCifsBranchCache	Removes the BranchCache service	C
Set-NcCifsBranchCache	Modifies the CIFS BranchCache service settings	C
Set-NaCifsBranchCacheKey	Sets the server secret for BranchCache	7

Table 25 lists the commands for Symlink and character map management. Symlinks are used to allow an SVM or a 7-Mode controller to map a file system to a CIFS-type share and an NFS export at the same time. The commands to modify character maps allow invalid file name characters to be remapped to valid file names. These Symlink and character map settings are outside the scope of PowerShell. However, the commands exist to fully manage both of these scenarios.

In Table 25, C represents clustered Data ONTAP.

Table 25) Commands for Symlink and character map management.

Command	Function	Available In
Add-NcCifsSymlink	Creates a new CIFS symbolic link path mapping from a UNIX Symlink to a target CIFS path	C
Get-NcCifsSymlink	Gets a list of CIFS symbolic link path mappings	C
Set-NcCifsSymlink	Sets CIFS symbolic link path mapping parameters	C
Remove-NcCifsSymlink	Removes a CIFS symbolic link path mapping	C
New-NcCifsCharacterMapping	Creates a character mapping	C
Get-NcCifsCharacterMapping	Gets CIFS character mapping configured for volumes	C
Set-NcCifsCharacterMapping	Modifies a character mapping	C
Remove-NcCifsCharacterMapping	Deletes a character mapping	C

4.9 NFS Commands

In a method similar to the commands for CIFS (SMB) shares, the steps in this section are outlined to create an NFS export start with enabling the service. Run the following commands to add and enable the NFS services to a 7-Mode controller or a clustered Data ONTAP SVM.

```
PS:> Enable-NaNfs # 7Mode command to start the NFS service
PS:> Add-NcNfsService # CDOT command to start the NFS service
```

The NFS service is set to default values; however, clustered Data ONTAP offers a number of user-customizable settings that enable specific NFS versioning and supportability options.

You can change the settings by running the `set-NcNFSSettings -attribute value` command and then using the attributes listed in Table 26.

Note: Table 26 lists only the most commonly used configuration settings for an NFS service; it is not an exhaustive list. To generate a complete list, run the PowerShell command `get-help set-NcNfsService -detailed`.

Table 26) Commonly used configuration settings for an NFS service.

Attributes	
ChownMode	IsNfsV3ChangeEnabled
DefaultWindowsUser	IsNfsV4ChangeEnabled
EnableEJukebox	IsNfsv40AclEnabled

Attributes	
IsNfsV2Enabled	IsNfs41AclEnabled
IsNfsV3Enabled	IsNfsV3ConnectionDropEnabled
IsNfsV40Enabled	NTFSUnixSecurityOps
IsNfsV41Enabled	NfsV4IdDomain
IsNfsV4PnfsEnabled	

Table 27 lists the set of commands that enable you to set up and configure an NFS service. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 27) Commands for setting up and configuring an NFS service.

Command	Function	Available In
Add-NcNfsService	Creates a new NFS configuration	C
Disable-N?Nfs	Stops the NFS service	7, C
Enable-N?Nfs	Starts the NFS service	7, C
Get-NcNfsService	Gets the NFS server configuration	C
Remove-NcNfsService	Deletes an NFS configuration	C
Set-NcNfsService	Modifies an NFS configuration	C
Test-NcNfs	Gets the status of the NFS server, whether or not it is running	C

Starting the NFS service doesn't expose an export to a client. To complete the export process, run the following command:

```
PS:> Add-NcNfsExport /NFSVol -ReadWrite all-hosts -NoSuid -SecurityFlavors sys
```

Table 28 list the commands to export an NFS share. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 28) Commands for exporting an NFS share.

Command	Function	Available In
Add-N?NfsExport	Enables path names for mounting according to the rules specified	7, C
Get-N?NfsExport	Gets the rules associated with exports	7, C
Remove-NcNfsExport	Removes the rules for a set of path names	C
Set-NcNfsExport	Similar to the Add-NcNfsExport command except it returns with an error if the rule does not already exist	C

Table 29 lists the commands to monitor statistics for the NFS service. These commands exist only for 7-Mode controllers because clustered Data ONTAP has an improved method for gathering statistics using performance commands.

```
PS:> Get-NaNfsStatistics # Retrieves the NFS counters for a 7-Mode Controller
PS:> Get-NcPerfObject # Retreives list possible objects to monitor, including NFS ones
```

Note: See the help files for examples of how to gather this performance information about a clustered Data ONTAP SVM.

In Table 29, 7 represents Data ONTAP operating in 7-Mode.

Table 29) Commands for monitoring statistics for the NFS service.

Command	Function	Available In
Add-NaNfsMonitor	Starts monitoring the specified hosts for NFS lock recovery purposes	7
Clear-NaNfsStatistics	Sets all NFS statistics to zero	7
Get-NaNfsMonitor	Lists the hosts that are currently being monitored by the NFS status monitor	7
Get-NaNfsStatistics	Collects NFS statistics for a specified client	7
Get-NaNfsTopClient	Returns a list of the top NFS clients, ordered by total NFS operations	7

Table 30 lists the advanced commands that allow you to either troubleshoot or fine-tune an NFS export or the NFS service. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 30) Commands for troubleshooting and fine-tuning an NFS export or NFS service.

Command	Function	Available In
Clear-NaNfsExportCache	For the given path or all paths, renews or flushes the access cache	7
Clear-NaNfsLock	Reclaims NFS locks	7
Disable-NaNfsExportFence	Disables fencing to the given exports for the given entry	7
Enable-NaNfsExportFence	Enables fencing to the given exports for the given entry	7
Get-NaNfsExportStoragePath	For the given path, determines the actual storage path	7, C
Get-N?NfsSecurityFlavor	Gets a list of currently supported security types	7, C
Test-NcNfsExportPermission	Tests if the host IP has mount permissions for a specified path	C

4.10 Deduplication Commands

NetApp deduplication is controlled and configured by FlexVol software, which allows granular control while also allowing control over which LUNs, shares, or exports can be deduplicated against each other.

The steps to enable deduplication on a volume are to first enable the single-instance store (SIS) feature on the volume, configure the SIS policy for the volume (optional), and start the initial SIS scan. You need to know the volume that you want to deduplicate and the policy regarding when to run the deduplication process.

The following script is an example of running the output of the `Get-NcVol` command and using a filter to pipeline on which volumes to enable SIS:

```
PS:> Get-NcVol -vserver infra | Where-Object{$_ .size -gt 100g} | Enable-NcSIS
# Enables the SIS to the Volume

PS:> Set-NcSIS Volume3 -Schedule "mon-fri@0,12"
# Sets up the automated policy. M-T-W-T-F at Noon and Midnight

PS:> Get-NcSIS Volume3 | Start-NcSIS
```

The commands let you define different sets of policies for deduplication that can then be applied to volumes by name. For example, you might create a deduplication policy called `HyperActive`, which runs four times a day, seven days a week, and might affect the volume's performance. Alternately, you might create another schedule called `SuperPassive`, which runs only on Sundays at 2 a.m., runs without affecting user load, and is limited to running no more than six hours at a time.

```
PS:> New-NcSisPolicy -name HyperActive -schedule "daily@0,6,12,18" -QosPolicy Best-Effort
PS:> New-NcSisPolicy -name SuperPassive -schedule "sun@2" -QosPolicy background
```

To set this policy on a specific volume called `Large`, run the following command:

```
PS:> Get-NcSIS Large | Set-NcSis -policy SuperPassive
```

Table 31 lists the complete list of deduplication (SIS) commands. In this table, 7 represents Data ONTAP operating in 7-Mode and C represents clustered Data ONTAP.

Table 31) Complete list of deduplication commands.

Command	Function	Available In
<code>Disable-N?Sis</code>	Disables SIS (dedupe) on a volume	7, C
<code>Enable-N?Sis</code>	Enables SIS (dedupe) on a volume	7, C
<code>Get-N?Sis</code>	Gets the SIS (dedupe) status for one or more volumes	7, C
<code>Get-NcSisPolicy</code>	Gets the SIS (dedupe) policies defined on one or more SVMs	C
<code>New-NcSisPolicy</code>	Creates a new SIS (dedupe) policy	C
<code>Remove-NcSisPolicy</code>	Deletes a SIS (dedupe) policy	C
<code>Set-N?Sis</code>	Sets up or modifies SIS (dedupe) policy, schedule, or options for a volume	7, C
<code>Set-NcSisPolicy</code>	Modifies the attributes of a SIS (dedupe) policy	C
<code>Start-N?Sis</code>	Starts a SIS (dedupe) operation on a volume	7, C
<code>Stop-N?Sis</code>	Aborts any currently active SIS (dedupe) operation on the volume	7, C

5 PowerShell Commands for SANtricity (E-Series)

Using PSTK for controllers based on NetApp SANtricity® software has a dependency different than that of the other PowerShell modules. Specifically, the controllers need the NetApp SANtricity web services proxy agent running somewhere in the infrastructure to allow the PowerShell commands to be sent.

PSTK for SANtricity can be loaded by running the following command:

```
PS:> Import-Module NetApp.SANtricity.PowerShell
```

After the web proxy agent is installed, run the following command to connect to the web proxy:

```
PS:> $URL = "https://127.0.0.1:8080/devmgr" #use the IP address of your Web Proxy Agent here
PS:> $HN = "YourHostName" #insert your hostname here
PS:> New-NeCredential -CredentialUser admin -Credentialname $hn -proxyurl $url
```

A warning regarding PSTK using unapproved verbs is displayed. This warning does not affect the ability to use any commands or block the use of any commands. There are currently over 250 commands in the PSTK that relate to SANtricity controllers. These commands are broken into the sectioned categories shown in Table 32.

The SANtricity PSTK lacks a current list of help files. Table 32 shows which commands are available; common commands can be discovered using the PowerShell integrated scripting environment (ISE).

The command categories help you to determine which commands can be used for which purpose.

Table 32) PSTK command categories for SANtricity controllers.

Category	Command	Description
Configuration	Get-NeConfig	Retrieves the current result data for the last operation; if no operation has been performed, an empty body is returned.
	Get-NeConfigValidate	Retrieves a list of known configuration items that can be used by this server.
	Remove-NeConfig	Interrupts any current configuration process.
	Update-NeConfig	Starts a new bulk configuration operation. If an operation is already running, a 200 is returned. If a new operation started, a 201 is returned.
	Update-NeConfigValidate	Validates an input CSV file previously uploaded.
Consistency groups	Basic	
	Get-NeCG	Retrieves all consistency groups
	New-NeCG	Creates a consistency group
	Remove-NeCG	Removes a consistency group
	Update-NeCG	Updates a consistency group
	Member Volumes	
	Get-NeCGMemberVolumes	Retrieves consistency group members
	Remove-NeCGMemberVolumes	Removes consistency group members

Category	Command	Description
	Update-NeCGMemberVolumes	Adds a consistency group member
	Update-NeCGMemberVolumesBatch	Adds multiple consistency group members at once
	Views	
	Get-NeCGViews	Retrieves Snapshot copy views associated with the consistency group
	Get-NeCGViewsViews	Retrieves the underlying Snapshot copy volumes associated with the <code>PITConsistencyGroupView</code> command
	New-NeCGViews	Creates Snapshot copy views from a consistency group Snapshot copy
	New-NeCGViewsBatch	Creates Snapshot views from a consistency group Snapshot copy
	Remove-NeCGViews	Deletes an existing Snapshot view
Snapshot copies	Basic	
	Get-NeSnapshotImages	Retrieves all Snapshot copies
	New-NeSnapshotImages	Creates a Snapshot copy
	Remove-NeSnapshotImages	Deletes a Snapshot volume
	Consistency Groups	
	Get-NeCGSnapshots	Retrieves Snapshot copies associated with the consistency group
	New-NeCGSnapshots	Creates a consistency group Snapshot copy
	Remove-NeCGSnapshots	Removes a consistency group Snapshot copy
	Update-NeCGSnapshotsRollback	Rolls back a consistency group Snapshot copy
	Legacy	
	Get-NeLegacySnapshot	Retrieves legacy Snapshot copies
	Get-NeLegacySnapshotRepositoryUtilization	Retrieves the repository usage statistics for a single legacy Snapshot copy
	New-NeLegacySnapshot	Creates a legacy Snapshot copy
	Remove-NeLegacySnapshot	Deletes a legacy Snapshot copy
	Update-NeLegacySnapshot	Updates a legacy Snapshot copy
	Groups	
	Get-NeSnapshotGroup	Gets all Snapshot copy groups
	Get-NeSnapshotGroupRepositoryUtilization	Retrieves the repository usage statistics for a single Snapshot copy group

Category	Command	Description
	New-NeSnapshotGroup	Creates a Snapshot copy group
	Remove-NeSnapshotGroup	Deletes a Snapshot copy group
	Update-NeSnapshotGroup	Updates a Snapshot copy group
	Volumes	
	Get-NeSnapshotVolume	Retrieves a Snapshot copy volume
	Get-NeSnapshotVolumeRepositoryUtilization	Retrieves the repository usage statistics for all Snapshot copy volumes
	New-NeSnapshotVolume	Creates a Snapshot copy volume
	Remove-NeSnapshotVolume	Deletes a Snapshot copy volume
	Update-NeSnapshotVolume	Updates a Snapshot copy volume
	Update-NeSnapshotVolumeConvertReadOnly	Converts a read-only Snapshot copy volume to read-write mode
	Schedule	
	Get-NeSnapshotSchedules	Retrieves all Snapshot copy schedules
Repositories	Get-NeRepositoriesConcat	Retrieves the list of concat repository volumes
	Update-NeRepositoriesConcatExpand	Expands a concat repository volume by providing an expansion candidate
	Update-NeRepositoriesConcatMultiple	Requests the best concat volume candidate for multiple base volumes
	Update-NeRepositoriesConcatSingle	Requests a list of concat volume candidates for a single base volume
Diagnostics	Get-NeDeviceAlerts	Retrieves the device alerts configuration
	Get-NeDeviceAlertsAlertSyslog	Retrieves the syslog configuration
	Test-NeDeviceAlertsAlertEmailTest	Initiates a test e-mail by using the array alert settings
	Test-NeDeviceAlertsAlertSyslogTest	Initiates sending test syslog messages by using the syslog settings
	Update-NeDeviceAlerts	Updates the device alerts configuration
	Update-NeDeviceAlertsAlertSyslog	Updates the syslog configuration
	Update-NeDiagnosticData	Returns various data about the managed device for diagnostic or display purposes
	Update-NeSupportData	Initiates a support data retrieval request
	Get-NeSupportData	Retrieves the status of a pending data retrieval request
File management	Get-NeFiles	Retrieves a file from the scratch directory

Category	Command	Description
	Remove-NeFiles	Deletes a generic file
	Update-NeFiles	Uploads a generic file for later reference in an API call
Firmware	Get-NeCfwUpgrade	Retrieves the status of a controller firmware upgrade operation
	Get-NeFirmwareCfwFiles	Retrieves the list of firmware files
	Get-NeFirmwareCompatibilityCheck	Requests the status of a firmware compatibility check operation
	Remove-NeFirmwareCompatibilityCheck	Cancels a firmware compatibility check operation
	Remove-NeFirmwareUpload	Deletes an uploaded firmware file
	Update-NeCfwUpgrade	Initiates a controller firmware upgrade operation
	Update-NeCfwUpgradeActivate	Activates staged controller firmware
	Update-NeFirmwareCompatibilityCheck	Begins a firmware compatibility check operation
	Update-NeFirmwareUpload	Uploads a firmware file
Flash Cache	Get-NeFlashCache	Retrieves NetApp Flash Cache™ volume if it exists
	Get-NeFlashCacheCompatibleVolumes	Retrieves a list of volumes that are compatible with the defined Flash Cache volume
	New-NeFlashCache	Defines a new Flash Cache volume
	Remove-NeFlashCache	Deletes the defined Flash Cache volume
	Set-NeFlashCacheResume	Resumes a suspended Flash Cache volume
	Set-NeFlashCacheSuspend	Suspends the Flash Cache volume
	Update-NeFlashCacheAddDrives	Adds drives to an existing Flash Cache volume
	Update-NeFlashCacheConfigure	Modifies Flash Cache parameters
	Update-NeFlashCacheRemoveDrives	Removes drives currently being used by Flash Cache
Hardware	Basic	
	Get-NeController	Retrieves a controller
	Get-NeDrive	Retrieves a drive
	Get-NeDriveUnreadableSectors	Retrieves the list of unreadable sectors
	Update-NeDrive	Selects drives for storage pool creation

Category	Command	Description
	Get-NeHardwareInventory	Retrieves hardware information
	Get-NeHardwareInventoryConnections	Retrieves connectivity information for drive trays
	Remove-NeIdentify	Cancels any active hardware identification
	Update-NeIdentify	Starts hardware identification
	iSCSI	
	Get-NeIscsiEntity	Retrieves the iSCSI entity data
	Get-NeIscsiTargetSettings	Retrieves the iSCSI target
	Update-NeIscsiTargetSettings	Updates the iSCSI target
Health check	Get-NeHealthCheck	Retrieves health check status
	Remove-NeHealthCheck	Cancels a running health check
	Update-NeHealthCheck	Begins a health check
Host groups	Get-NeHostGroup	Retrieves all host groups
	New-NeHostGroup	Creates a host group
	Remove-NeHostGroup	Deletes a host group
	Update-NeHostGroup	Updates a host group
	Get-NeHost	Retrieves a host
	Get-NeHostPortTypes	Retrieves a host type
	Get-NeHostTypes	Retrieves all host types
	Get-NeHostTypeValues	Retrieves the list of host types
	Get-NeUnassociatedHostPorts	Retrieves the list of unassociated host ports
	New-NeHost	Updates the parameters of a host
	Remove-NeHost	Deletes a host
	Update-NeHost	Updates the parameters of a host
	Update-NeHostMove	Moves a host under a different host group
	Update-NeHostPortTypes	Sets a default host type
	Update-NeHostTypes	Sets a default host type
Mirroring	Async	
	Get-NeAsyncMirrorGroup	Retrieves async mirror groups
	Get-NeAsyncMirrorGroupArvmArrays	Retrieves a list of storage systems that support async mirroring

Category	Command	Description
	Get-NeAsyncMirrorGroupConnections	Retrieves the current remote storage device mirroring connections for an async mirror group
	Get-NeAsyncMirrorGroupIncompletePairs	Retrieves async mirror group relationships that have not yet been completed
	Get-NeAsyncMirrorGroupPairs	Retrieves all async mirror group members
	Get-NeAsyncMirrorGroupPairsRepositoryUtilization	Retrieves the repository usage statistics for all async mirror group members
	Get-NeAsyncMirrorGroupProgress	Retrieves the sync progress of all async mirror groups at once
	New-NeAsyncMirrorGroup	Creates an async mirror group
	New-NeAsyncMirrorGroupPairs	Adds a new member to an async mirror group
	Remove-NeAsyncMirrorGroup	Deletes an async mirror group
	Remove-NeAsyncMirrorGroupIncompletePairs	Removes an async mirror relationship that has not been completed
	Remove-NeAsyncMirrorGroupPairs	Removes a member from async mirror group
	Set-NeAsyncMirrorGroupResume	Resumes sync of an async mirror group
	Set-NeAsyncMirrorGroupSuspend	Suspends sync of an async mirror group
	Set-NeAsyncMirrorGroupSync	Begins sync of an async mirror group
	Test-NeAsyncMirrorGroupTest	Tests connectivity of an async mirror group
	Update-NeAsyncMirrorGroup	Updates an async mirror group
	Update-NeAsyncMirrorGroupIncompletePairs	Completes an incomplete async mirror group relationship
	Update-NeAsyncMirrorGroupRole	Updates the mirror role of an async mirror group
	Remote	
	Get-NeRemoteMirrorPairs	Retrieves remote volume mirror
	Get-NeRemoteMirrorPairsMirrorSyncProgresses	Retrieves remote mirror sync status
	Get-NeRemoteMirrorPairsRemoteMirrorTargetCandidates	Retrieves potential mirror candidates for a volume
	New-NeRemoteMirrorPairs	Creates a remote volume mirror
	Remove-NeRemoteMirrorPairs	Removes a remote volume mirror

Category	Command	Description
	Test-NeRemoteMirrorPairsTestRemoteMirrorCommunication	Tests remote mirror communication
	Update-NeRemoteMirrorPairs	Updates a remote volume pair
Monitoring	Get-NeEvents	Retrieves all global status events
	Get-NeMelEvents	Retrieves MelEvents
	Get-NeMelEventsAvailable	Checks the oldest and newest available events
	Remove-NeMelEvents	Clears MelEvents
Service catalog	Get-NeSscPools	Retrieves the list of pools
	Get-NeSscVolume	Retrieves a volume by ID or label
	New-NeSscVolume	Defines a new volume and configures its QoS parameters
	Remove-NeSscVolume	Deletes a volume by ID or label
	Update-NeSscVolume	Updates an existing volume
Statistics	Get-NeAnalysedDriveStatistics	Retrieves all analyzed disk statistics
	Get-NeAnalysedVolumeStatistics	Retrieves analyzed volume statistics
	Get-NeDriveStatistics	Retrieves raw disk statistics
	Get-NeVolumeStatistics	Retrieves raw volume statistics
Storage system	Get-NeDiscovery	Retrieves discovery results
	Get-NeFolder	Retrieves all folders
	Get-NeFolderStorageDevices	Retrieves storage devices associated with a folder
	New-NeFolder	Creates a folder
	Remove-NeFolder	Deletes a folder
	Update-NeFolder	Updates a folder
	Get-NeStorageSystem	Retrieves a specific storage system
	New-NeStorageSystem	Adds a storage system
	Remove-NeStorageSystem	Removes one or more storage systems
	Update-NeStorageSystem	Updates a storage system
	Get-NeCapabilities	Retrieves storage device capabilities
	Get-NeConfigEthernetInterfaces	Retrieves the list of Ethernet interfaces
	Update-NeConfigEthernetInterfaces	Configures the Ethernet management connections on a controller

Category	Command	Description
	Get-NeGraph	Retrieves the object graph, which contains all configuration details for the storage system
	Get-NePassword	Retrieves the password status of the storage device
	Update-NePassword	Sets the password of the storage device
	Remove-NeDiscovery	Cancels the discovery operation
	Update-NeDiscovery	Discovers storage devices on the network
	Update-NeValidatePassword	Validates the stored password for a storage system
Upgrade	Get-NeUpgrade	Returns version information for the web services proxy software that is currently running (same as /utils/about) and the version of any staged updates; if there are no updates, the array of version data is empty
	Update-NeUpgradeDownload	Starts a download of software updates from the update server to the staging area. This operation runs asynchronously. A set of events are posted to the event queue devmgr/v2/events that indicate the status of the process.
	Update-NeUpgradeReload	Starts a reload of the software. If any updates are downloaded, they are loaded. This operation runs asynchronously. A set of events are posted to the event queue devmgr/v2/events that indicate the status of the process.
Volumes	Basic	
	Get-NeVolume	Retrieves a specific volume
	New-NeVolume	Creates a volume
	Remove-NeVolume	Deletes a volume
	Update-NeVolume	Updates volume parameters
	Update-NeVolumeExpand	Starts the volume expansion
	Update-NeVolumeInitialize	Initializes a volume
	Get-NeAccessVolume	Retrieves the access volume
	Get-NeMappableObjects	Retrieves a list of all mappable objects
	Volume Mappings	
	Get-NeVolumeMappings	Retrieves the list of LUN mappings
	New-NeVolumeMappings	Creates a new LUN mapping
	Remove-NeVolumeMappings	Removes a LUN mapping

Category	Command	Description
	Update-NeVolumeMappingsMove	Moves a LUN mapping to a different host or host group
	Volume Group	
	Update-NeVolumeGroupRaidTypeMigration	Performs a RAID-type migration on a storage pool
	Update-NeVolumeGroupReduction	Reduces the number of drives on a storage pool
	Get-NeVolumeExpand	Expands the capacity of a storage pool
	Get-NeVolumeGroupActionProgress	Checks the progress of a long-running action on a storage pool
	Get-NeVolumeGroup	Retrieves a storage pool
	Get-NeVolumeGroupExpand	Retrieves a list of expansion candidates for a storage pool
	Get-NeVolumeGroupReduction	Reduces the number of drives on a storage pool
	New-NeVolumeGroup	Creates a storage pool
	Remove-NeVolumeGroup	Deletes a storage pool
	Update-NeVolumeGroup	Updates a storage pool
	Update-NeVolumeGroupExpand	Expands the capacity of a storage pool
	Thin Volume	
	Get-NeThinVolume	Retrieves a thin volume
	New-NeThinVolume	Creates a thin volume
	Remove-NeThinVolume	Deletes a thin volume
	Update-NeThinVolume	Updates a thin volume
	Update-NeThinVolumeExpand	Expands a thin volume
	Update-NeThinVolumeInitialize	Initializes a thin volume
	Volume Copy Jobs	
	Get-NeVolumeCopyJobs	Retrieves the list of volume copy pairs
	New-NeVolumeCopyJobs	Creates a new volume copy pair
	Remove-NeVolumeCopyJobs	Removes a volume copy pair
	Update-NeVolumeCopyJobs	Updates parameters of a volume copy pair

6 Advanced User Permissions

Using E-Series storage systems, it is not possible to define levels of control. However, it is possible to set up a granular level of control over commands in PSTK. This level of control is enforced by the NetApp FAS controllers.

The first step to set up advanced user permissions is to create a user name and password (or group) on the NetApp controller for the user or group that has limited access. You must determine which commands you want this user to be able to execute and the required API access needed to accomplish these commands.

By using the help information for each command or the `Show-NCHelp` command, using the API tab you can discover the API access required for each command. For example, if you want to allow a user to run the `Get-NcVol` command, that user would need access (permission) to use the API called `Volume-Get-Iter`. If you grant access to the `Volume-Get-Iter` API, then the other commands that rely on that API, such as `Get-NcVolContainer`, `Get-NcVolRoot`, and `Test-NcSnapMirrorVolume`, also work.

As an example, Table 33 shows the API calls that are required to execute PowerShell commands related to controller licenses.

Table 33) API calls required to execute PowerShell commands related to controller licenses.

API Requirement	PowerShell Command
license-v2-add	Add-NcLicense
license-v2-delete	Remove-NcLicense
license-v2-delete-expired	Remove-NcLicense
license-v2-delete-unused	Remove-NcLicense
license-v2-entitlement-risk-get-iter	Get-NcLicenseEntitlementRisk
license-v2-list-info	Get-NcLicense

Note: A single command, such as the `Remove-NcLicense` command, might require access to a family of APIs. You can control the behavior of the PowerShell command to allow only the removal of expired or unused licenses by granting access to the appropriate licenses while denying the ability to remove a currently applied valid license.

As shown in Table 34, it is also possible to run multiple commands by granting a single API.

Table 34) Example of single API running multiple commands.

API	Commands
volume-get-iter	Get-NcVol Get-NcVolContainer Get-NcVolRoot Test-NcSnapmirrorVolume

By granting access to the API that allows the `Get-NcVol` command to run, you are also granting access for the similar `Get-NcVolRoot` command because that command uses the same API mechanisms.

The list of APIs exists as a tree structure and, by default, an administrative account has access to the entire tree. You can, however, pick and choose which branches are accessible to a user.

Table 35 shows that you can grant access to the following APIs to manage a cluster but deny the ability to reset or reboot nodes.

Table 35) Direct system control APIs.

API List	Command List	Allow Access?
system-api-list	Get-NcSystemApi	Yes
system-cli	Disable-NcNdmp Enable-NcNdmp Set-NcTime	No
system-get-node-info-iter	Get-NcNodeInfo	Yes
system-get-ontapi-version	Get-NcSystemOntapiVersion	Yes
system-get-vendor-info	Get-NcSystemVendorInfo	Yes
system-get-version	Get-NcSystemVersion Get-NcSystemVersionInfo	Yes
system-image-fetch-package	Start-NcSystemImagePackageDownload	Yes
system-image-get-iter	Get-NcSystemImage	Yes
system-image-install	Start-NcSystemImageInstall	Yes
system-image-modify	Set-NcSystemImage	Yes
system-image-package-delete	Remove-NcSystemImagePackage	Yes
system-image-package-get-iter	Get-NcSystemImagePackage	Yes
system-image-update	Start-NcSystemImageUpdate	Yes
system-image-update-get-abort	Stop-NcSystemImageUpdateGet	Yes
system-image-update-progress-get	Get-NcSystemImageUpdateStatus	Yes
system-node-discovery-get-iter	Get-NcNode	Yes
system-node-get-iter	Get-NcNode	Yes
system-node-modify	Set-NcNode	No
system-node-power-cycle	Restart-NcNode	No
system-node-power-get	Get-NcNodePower	Yes
system-node-power-off	Stop-NcNode	No
system-node-power-on	Start-NcNode	Yes
system-node-reboot	Restart-NcNode	No
system-node-rename	Rename-NcNode	Yes
system-node-reset	Reset-NcNode	No
system-node-shutdown	Stop-NcNode	No

API List	Command List	Allow Access?
system-services-web-get	Get-NcSystemWebServices	Yes
system-user-capability-get-iter	Get-NcSystemUserCapability	Yes

You can use wildcards (*) to identify unlimited access to an entire branch or only to a subbranch.

Table 36 shows how the list of required APIs can be drastically shortened by using wildcards to accomplish the same task without having to specify each API.

Table 36) Wildcard-enabled permissions list.

API List		
System-node-rename	System-node-power-on	System-node-power-get
System-Services-*	System-node-get-*	System-User-*
System-Image-*	System-Get-*	System-node-discover-*

To grant access to all of the system commands, use the API set `System-*`.

To determine the API access needed for a specific command, run the following command from a PowerShell prompt:

```
PS:> Get-NcHelp Get-NcVol | select API
```

To create a nonadmin role on the controller, start with the APIs listed in Table 37.

Table 37) Suggested APIs for nonadmin role access on controller.

API	Description
api-aggr-*	Allows full manipulation of Data ONTAP aggregates, including those embedded in traditional volumes and their structural components (plexes and RAID groups).
api-ems-*	The event management system (EMS) is a mechanism in the Data ONTAP kernel that supports creation, forwarding, and consumption of event indications. EMS events are generated by Data ONTAP when errors occur or to log changes in the status of the system. These event indications are logged to <code>/etc/log/ems</code> and, depending on the event and its severity, to the syslog console. If an event has an SNMP definition, SNMP traps are also generated by the event indication.
api-fcp-adapter-*	Grants read and write permissions to all API-initiated FCP adapter operations: configuration, reset, and up and down status.
api-igroup-*	Grants read and write permissions to all API-initiated igroup operations: add, bind, create, destroy, remove, rename, and so on.
api-iscsi-*	Grants read and write permissions to all API-initiated iSCSI operations: service start and service stop, disable, destroy, create, enable, and so on.
api-license-*	Grants read and write permissions to all API-initiated license operations: add, delete, and list.
api-lun-*	Grants read and write permissions to all API-initiated LUN operations: create, destroy, online, offline, and so on.

API	Description
api-qtree-*	Grants read and write permissions to all API-initiated qtree operations: create, destroy, list, rename, and so on
api-snapmirror-*	Grants read and write permissions to all API-initiated SnapMirror operations: service on, service off, break, abort, initialize, and so on
api-snapshot-*	Grants read and write permissions to all API-initiated Snapshot copy operations: create, delete, rename, restore volume, restore volume, and so on
api-snapvault-*	Grants read and write permissions to all API-initiated SnapVault operations: adding and removing relationships, schedule modifications, abort, create, restore, and so on
api-system-*	Grants read permissions to all API-initiated system operations: get version, get info, and so on
api-volume-*	Grants read and write permissions to all API-initiated volume operations: create, destroy, online offline, verify, split, restrict, size, and so on
login-*	Allows the account to log in using Telnet, SSH, rsh, console, and HTTP/S

Run the `Invoke-SSH` command and the following command to build the nonadmin role on the controller:

```
"useradmin role add MyRole-non-admin -a login-*,api-lun-*,api-snapshot-*,api-iscsi-*,api-volume-*,api-snapmirror-*,api-snapvault-*,api-ems-*,api-igroup-*,api-qtree-*,api-fcp-adapter-*,api-license-*,api-system-*,api-aggr-*
```

After the role is defined, run the `Invoke-SSH` command again to assign the role to a group.

```
"useradmin group add NonAdminGroup -r MyRole-non-admin"
```

After the group is set with the proper role, any user that is a member of the group has restricted role access to the NetApp controllers.

7 Integration with Additional Windows Features

All tasks in Windows 2012 and later can be performed by using PowerShell. Gather a list of available PowerShell modules by running the following command:

```
PS:> Get-Module -listavailable
```

For example, run the following commands to import a module, such as the iSCSI module, and then determine all of the available commands.

```
PS:> Import-module iscsi
PS:> (Get-module iscsi).exportedcommands
```

Use the same process to access the help document for these commands.

```
PS:> Get-Help Get-iSCSITarget -full
```

7.1 Event Logging

PowerShell scripts use a concept called verbose mode, which can be enabled in your scripts and by using a passed parameter. The following example shows how to use a Boolean value in your script that can be tested to determine if a verbose message should be displayed.

```
param([switch]$verbose)
```

Within the code body of the script, you can insert a sample output that only is displayed if verbose mode is enabled by running the `passed` command.

```
If ($verbose) {Write-host "This message only displayed since verbose mode is on"}
```

Similarly, you can use the `-WhatIf` argument at the end of the command to test the command and display the actions of the command without actually executing the command. Another commonly passed variable is the `Forced` argument, which causes an action to occur despite a script's built-in safety checks to abort specific actions based on detected scenarios.

It is good practice when managing an infrastructure to log all changes that occur to help determine the root cause when troubleshooting an infrastructure. This logging recommendation, as well as a strong change management requirement, makes scripted activities more attractive than working with a conventional GUI. The script shows the intention of actions to issue against the controller and the output from the script should equally show the success or failure of those commands.

To simplify your scripts, deploy a function in your scripts to support all user output and offer support for event logging as well as support for the verbose option listed earlier.

The following function accepts a passed set of values, which include the text and severity of the event. These events are passed to a log file, screen output, and the event log. If the log file does not exist, it creates a new empty log file. If the event log does not have a topic concerning this script, the log automatically creates one. If the verbose option is set, all of the possible messages are sent; otherwise, informational-type messages are not displayed or logged.

```
function PostEvent([String]$textfield, [string]$eventtype) # Post Events to Log/Screen/EventLog
{ $outfile = "C:\mylogfiledir\logs\myscript.log"
  if (!(test-path $OUTFILE))
  { $suppress = mkdir c:\SANKit\Logs # If the log file doesn't exist, then create it.
  }
  if (!(test-path HKLM:\SYSTEM\CurrentControlSet\Services\Eventlog\application\MyScript) )
  { New-Eventlog -LogName Application -source MyScript # Event log doesn't exist, lets create it
    PostEvent "Creating Eventlog\Application\MyScript EventLog Source" "Warning"
  }
  else
  { switch -wildcard ($eventtype) # Lets color-code the events for the screen
    { "Info*"      { $color="gray" }
      "Warn*"     { $color="green" }
      "Err*"      { $color="yellow" }
      "Cri*"      { $color="red" $EventType="Error" }
      default     { $color="gray" }
    }
  }
  if (!(($verbose) -and ($eventtype -eq "Information"))) # Suppress informational events
  { write-host "- $textfield -foregroundcolor $color # output to interactive screen
    write-Eventlog -LogName Application -Source MyScript -EventID 1 -Message $TextField `
      -EntryType $eventtype -Computername "." -category 0 # output to EventLog
    $textfield | out-file -filepath $outfile -append # Output to the Log File
  }
}
```

Triggering AutoSupport

It is common practice to update the controller records immediately before and after changing any significant settings to a NetApp controller. This practice enables you (and NetApp Support) to identify adverse effects from an unintended change. It is important to show within the AutoSupport embedded message that the message is user generated and that the status is Test AutoSupport Only so that NetApp does not take service action on the controller.

```
Invoke-NcAutoSupport -Message "Test Autosupport ONLY : User Generated record" -Type "all"
```

Using PowerShell as Proof-of-Concept Tool

In many cases, writing a proof-of-concept tool for a process can be done entirely in PowerShell. After the process and results are validated, the conversion process to a more robust language such as C#, C++, or Java can occur. The PowerShell commands can be used to discover the correct API calls to make to convert snippets of PowerShell commands to working functions.

Discovering APIs for Commands

The process of discovering APIs is no different from the process of discovering API usage for the user rights and roles listed in section 6, "Advanced User Permissions." You can gather additional information from the PowerShell commands such as the object format that can be passed into an API and the response format of other API calls.

8 Useful Sample Scripts

This section provides a collection of useful scripts for an infrastructure administrator to have in a personal toolkit. These scripts deal with daily activities as well as verification and validation activities to make sure that configurations do not differ from the accepted standards.

8.1 AutoSupportChecker Script

The AutoSupportChecker script checks the AutoSupport settings on a controller and compares them to a controller that is already configured. This script supports either a verbose option or an execute option. By default, the script only describes the differences between the two controllers. If you select the verbose options, all of the AutoSupport settings are displayed, regardless of the values. If you select the execute option, the script modifies the settings on the target controller to match the source (gold) controller.

```
# This script will connect to a Storage Controller and retrieve all
# of the AutoSupport Settings. It will then connect to a second
# controller and compare those settings. The recommended changes will be shown.
param ([parameter(position=0)] $Argument )
$WorkingController = "10.58.94.164" # Source Controller, your Gold settings
$TargetController = "10.58.94.81" # Target Controller, the controller that needs to be fixed/set
$User = "PutYourUserNameHere"
$Pass = "PutYourPasswordHere"
$WCPass = ConvertTo-SecureString $Pass -AsPlainText -Force
$Wcred = New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList
$User,$WCPass
$TCPass = ConvertTo-SecureString $Pass -AsPlainText -Force
$Tcred = New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList
$User,$WCPass
Connect-NaController $WorkingController -Credential $Wcred
$Workops = get-naoption autosupport.*
Connect-NaController $TargetController -Credential $Tcred
$Targops = get-naoption autosupport.*
Write-host "-----"
if ($Argument -ne "Execute") # Not going to actually Execute, just inform
{ Write-host "Execute with quotes around it was not detected."
  Write-host "Showing changes needed to second controller to match the first controller."
}
if ($Argument -ne "Verbose")
{ Write-host "Verbose with quotes around it was not detected."
  Write-host "Showing All settings on Both controller"
}
Write-host "-----"
foreach ($item in $workops)
{ $workname=$item.name
  $workval =$item.value
  foreach($titem in $targops)
  { if ($titem.name -eq $item.name)
    { if ($titem.value -eq $item.value)
      { if ($Argument -eq "Verbose")
```


8.3 SANKit

SANKit software can be run on a server (or set of servers) remotely to prepare an environment so that it can load all SAN-based code in a repeatable manner (in a loop). This process is best suited for an environment that consists of thousands of servers in which all of the servers are expected to be fully patched and run only supported levels of BIOS, firmware, drivers, and hot fixes (patches).

Reboot Survival

The SANKit process is designed to upgrade as much as possible before requiring a reboot. A server is configured to automatically log on and restart after the logon. It is common for a server to rerun the SANKit until a successful exit code is returned. To avoid having the same updates made repeatedly, before any update is made, a check is run to see if that update is complete or if the package is installed. If it is, then the step is skipped. This process can take hours for a large server farm, but it more commonly happens simultaneously on hundreds of machines.

For example, if the IPAK consists of 10 steps, the list of stages for the IPAK might look like the following example.

Note: This example is an oversimplification of the process, but it illustrates the concept and design goals.

1. IPAK start. Set autologon = true. Set run once to rerun the SANKIT on reboot.
 - a. Complete steps 1–5, then allow reboot.
 - b. Autologon. Run IPAK start. Set autologon = true. Set run once to rerun the SANKit on reboot.
2. Test steps 1–5: already done, skipping.
 - a. Complete steps 6–8, then allow reboot.
 - b. Autologon. Run IPAK start. Set autologon = true. Set run once to rerun the SANKit on reboot.
3. Test steps 1–8: already done, skipping.
 - a. Complete steps 9–10, then allow reboot.
 - b. Autologon. Run IPAK start. Set autologon = true. Set run once to rerun the SANKit on reboot.
4. Test steps 1–10: already done, skipping.
 - a. SANKit is complete. Turn autologon = off and remove run once command.

SANKit Assumptions

If SANKit is run and does not have NetApp arrays at the other end of the FC wire, SANKit should report only the WWNs of the installed HBAs and not complete the installation process.

If SANKit detects HBAs and those HBAs detect a NetApp array, then it is assumed that SANKit for NetApp should be installed.

SANKit Installation Requirements

The SANKit installation procedure requires the following components:

- Its own PowerShell script and accompanying files to a local copy (C:\SANKIT)
- Emulex HBA driver\BIOS\firmware
- NetApp Windows Hardware Update Kit
- NetApp multipath I/O device-specific module
- Microsoft FCINFO tool from Microsoft Developer Network (needed to detect HBA information without loading driver stack)

SANKit Safety Features

SANKit uses safety features to avoid causing downtime. Some of these features can be overridden by uncommenting `$FORCENETAPP=$TRUE` at the start of the script.

Note: The features that are marked with an asterisk (*) can be overridden.

- If no HBAs are detected, then abort the SANKIT installation.
- If live LUNs exist on the HBA, then abort the HBA driver and BIOS.
 - Complete this step for each HBA.
 - If live LUNs exist on any HBA, prevent a reboot.
- * If a PowerPath or Navisphere agent or Navisphere CLI is installed, abort SANKIT.
- * If no NetApp target is detected, abort the WHUK and DSM install.

SANKit also needs to offer the following silent installations if directed to by command line switches:

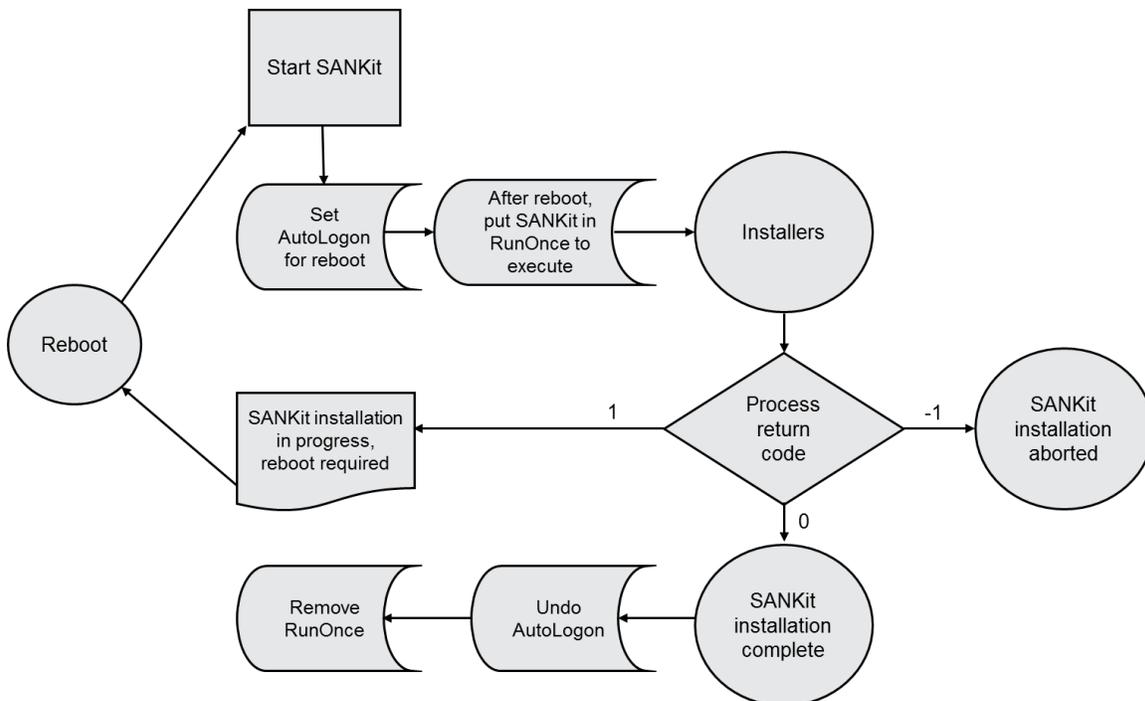
- NetApp SnapDrive® (/snapdrive) software
- MPIO DSM (/DSM)
- WHUK (/WINHUK)

SANKit Installation Process

You can run SANKit against a server that already hosts EMC LUNs. In this example, it is important to prevent an outage. The default behavior shown in Figure 2 avoids any risk on site while allowing the software installation.

Figure 2 is a workflow diagram of the SANKit installation process.

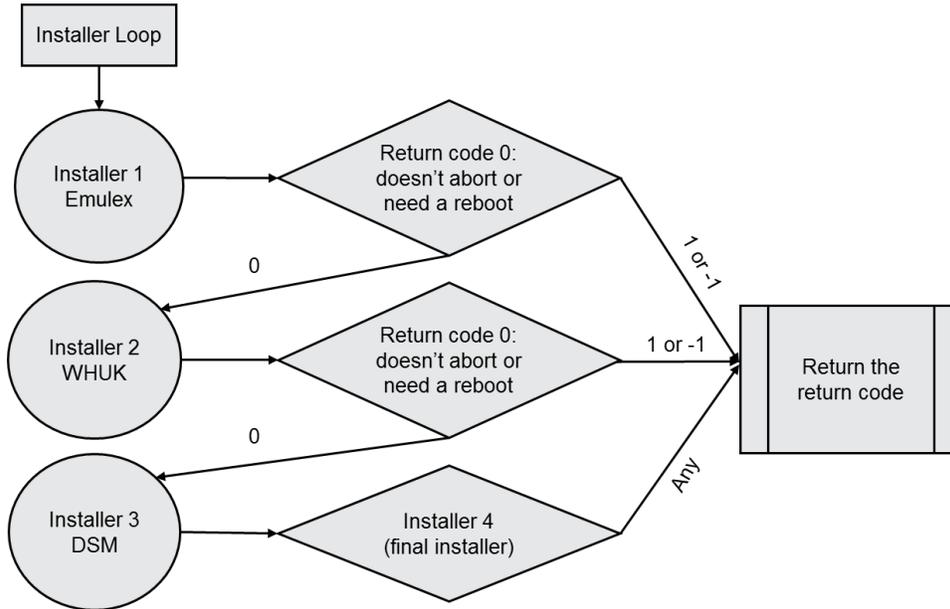
Figure 2) Flowchart showing the SANKit installation process.



Installer Loop

The overall SANKit installer is referred to as the installer loop. Figure 2 shows installer circles for the HBA, DSM, and Host Utilities Kit (HUK). The installers are modular and can be chained together. As shown in Figure 3, the chain of installers continues until an installer reboots or aborts or until all installers have completed.

Figure 3) Flowchart showing SANKit independent installation steps.



The reason for this unknown state, whether or not a reboot is required, is that the first time it enters the loop, installer 1 might load a driver that requires a reboot. However, the second time through, it discovers that the correct driver is installed so it returns successfully and goes on to installer 2. At this point, installer 2 runs and returns successfully, but it doesn't require a reboot and moves on to installer 3. Installer 3 might hit a problem and must abort the SANKit installation process. It can abort the installation process and return a failure back without being forced to either reboot or attempt to load installer 4, which shouldn't be loaded unless 3 is installed correctly.

Note: Between installer 2 and 4, a placeholder was left to install an optional component that is not defined in the current script.

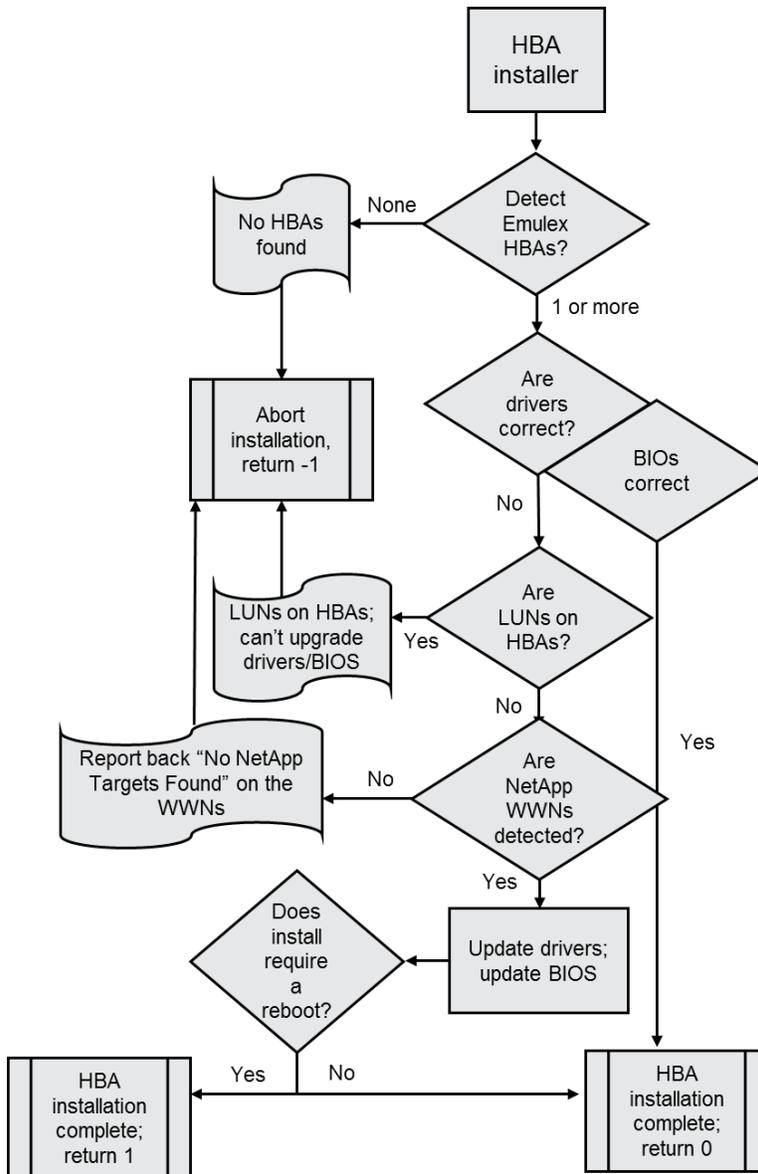
HBA Single Installer

Figure 4 shows a detailed look at the Emulex HBA single installer. There are three ways that this HBA single installer can abort SANKit:

- It fails to detect any HBAs.
- It finds the HBA driver/BIOS out of date but does not allow it to update because there are LUNs attached.
- It does not find NetApp targets on the fabric, which means it hasn't been zoned or connected properly.

Figure 4 also shows how the HBA single installer has one way to return the request for a reboot. If an update is performed, it is possible that a reboot is required before it completes. In this case, it has the option to give back control but demand a reboot. The last option is that the HBA installer completes successfully.

Figure 4) Flowchart showing decision matrix for SANKit.



The next steps, such as installing the HUK and the MPIO NetApp DSM, might require a reboot. These steps require similar tests to make sure that LUNs are not already mounted and a forced reboot doesn't occur on a functional server.

SANKit Runs and Validates Installation

If SANKit runs on a server that has all the correct drivers, BIOS, HUK, DSM, and so on, and the tools detect that the correct software is installed and then return back success, they do not require a reboot. SANKit can be run on any operational server without fear; therefore, it can be used to validate that all software is up to date and to install all new software.

Error Logging

Error logging occurs in several places (the event log is the default location). The IPAK creates a special Windows event log section called a hive in the IPAK event log where messages are stored. These

messages show NetApp as the SANKit source and they define information, warnings, and errors for several scenarios, such as:

- SANKit stop error
- Missing secondary path warning
- Existing installation information

For future scripts, create a hive to keep your events separate from the main application system log files.

Alternatively, direct the feedback back to the screen as well as in the command window. The feedback should be color coded to represent green (information), yellow (warning), orange (error), and red (critical).

If your code is called from another script, the code uses return codes to indicate success or failure.

Error Codes

Table 38 is a list of all possible SANKit error codes, which are ordered according to when they occur in the processing of the PowerShell script.

Note: The actions listed in Table 38 should be determined by the SANKit customer and considered the correct course of action for that site.

Table 38) SANKit error codes.

Error	Severity	Action
No HBAs found, exiting script.	Critical	Abort SANKit
LIVE LUNs detected on this HBA. Cannot update driver.	Critical	Continue
LIVE LUNs detected on this HBA. Cannot update firmware.	Critical	Continue
Nonsupported HBA model X running driver Y BIOS Z.	Critical	Continue
Exiting SANKit! No supported Emulex HBA found.	Critical	Abort SANKit
Installation of NetApp Windows Host Utility timed out.	Critical	Abort SANKit
EMC PowerPath has been detected. Aborting SANKit installation.	Critical	Abort SANKit
EMC Navisphere Agent has been detected. Aborting SANKit installation.	Critical	Abort SANKit
EMC Navisphere CLI has been detected. Aborting SANKit installation.	Critical	Abort SANKit
Installation of NetApp DSM timed out.	Critical	Abort SANKit
No NetApp target discovered; skipping DSM and WHUK install.	Critical	Skip WHUK-DSM
A reboot is required to complete the installation, but LUNs are online and reboot has been suppressed.	Error	Request user reboot
Forcing a reboot. Rerun the SANKit after the reboot to ensure that all packages installed correctly.	Error	Reboot server, rerun SANKit

If SANKit exits with any of the errors listed in Table 38, then reboot and rerun SANKit. If the same error occurs a second time, then SANKit will not install successfully. However, it is more likely that any open installation that might have blocked SANKit is no longer blocking or that the prescribed completion of SANKit required a reboot. In either case, it is safe to run SANKit again to verify that no errors are present.

SANKit Code Versions

Table 39 illustrates which code versions are installed with SANKit.

Table 39) SANKit code versions.

HBA Name	Version	File Name
Emulex HBA driver/HBAnyware	7.2.32.002	Elxocm-windows-x64-5.01.09.04-1.exe
Emulex LP111 BIOS	2.82A4	Ym282a4.all
Emulex LP1050 BIOS	2.82A4	Mb282a4.all
Emulex LP1150 BIOS	2.82A4	Wf282a4.all
Emulex LP1250 BIOS	2.82A4	Ob282a4.all
NetApp Windows Host Utilities Kit	5.3	Netapp_windows_host_utilities_5.3_x64.msi
NetApp DSM	3.4.1	Netapp_win_MPIO_3.4_setup_x64.msi

Note: The sample code supports many more models than are listed here. The extra models were added to support additional card types discovered during the testing phase because the insertion of these extra cards involved very low effort.

Sample Code

This section includes several methods of discovery as well as sample code to help with the logic of the scripts outlined in previous sections. The Fibre Channel Information Tool (fcinfo) (`FCINFO.exe`) must reside on the server. This tool allows you to interrogate any HBA that uses the HBA-API industry standard, which QLogic and Emulex both use. You can download and install the `FCINFO.exe` tool from the [Microsoft Fibre Channel Information Tool](#) download page.

The following examples show how to use the Fibre Channel Information Tool (fcinfo) (`FCINFO.exe`) to extract information about the HBAs and the fabric. Figure 5 shows the output of the `FCInfo` command, showing any valid HBAs in the machine.

Figure 5) Output from FCInfo command showing HBA information.

```

Administrator: Windows PowerShell Modules
PS C:\> fcinfo

There are 2 adapters:
  com.emulex-111-00454-0:      PortWWN: 10:00:00:00:c9:72:46:40      \\.\Scsi3:
  com.emulex-111-00454-1:      PortWWN: 10:00:00:00:c9:72:46:3f      \\.\Scsi4:
PS C:\>
  
```

Figure 6 shows the details about the installed HBAs, including the driver and firmware version.

Figure 6) Output from FCInfo command showing details about installed HBAs.

```

Administrator: Windows PowerShell Modules
PS C:\> fcinfo /details

adapter: com.emulex-111-00454-0
node_wwn: 20:00:00:00:c9:72:46:40
fabric: 10:00:00:05:1e:04:db:19
port_wwn: 10:00:00:00:c9:72:46:40
osdevice: \\.\Scsi3:
  vendor: x10DF
  prodid: xFE00
  nports: 1
  manfac: Emulex Corporation
  sernum: BG75146527
  model: 111-00454
  descrp: NetApp R6, SUB ASSY, HBA, 2P, 4G, FC, CLIENT
  symbic: Emulex 111-00454 FV2.72A2 DU7.2.30.016 HUUMDC0-03
  hwver: 2057706D
  driver: 7.2.30.016
  optver:
  fwver: 2.72A2
  drvnam: elxstor

adapter: com.emulex-111-00454-1
node_wwn: 20:00:00:00:c9:72:46:3f
fabric: 10:00:00:05:1e:02:02:07
port_wwn: 10:00:00:00:c9:72:46:3f
osdevice: \\.\Scsi4:
  vendor: x10DF
  prodid: xFE00
  nports: 1
  manfac: Emulex Corporation
  sernum: BG75146527
  model: 111-00454
  descrp: NetApp R6, SUB ASSY, HBA, 2P, 4G, FC, CLIENT
  symbic: Emulex 111-00454 FV2.72A2 DU7.2.30.016 HUUMDC0-03
  hwver: 2057706D
  driver: 7.2.30.016
  optver:
  fwver: 2.72A2
  drvnam: elxstor

PS C:\>
  
```

The `fcmap` command allows you to see all of the WWNNs that are connected to HBA0 or HBA1.

```
FCInfo /fcmap /ai:0
```

You can parse the output for NetApp branded WWNNs. You can also determine whether a device has a physical drive number mapped to it, as shown in Figure 7. There are maps in the list that do not have a physical drive number, indicating that the WWN is present but a LUN has not been assigned to this server.

Figure 7) Sample output from FCInfo showing detected NetApp target devices.

```

10.58.99.243 - Remote Desktop Connection
Administrator: Windows PowerShell Modules
PS C:\>
PS C:\> fcinfo /fcmap /ai:0

com.emulex-111-00454-0: num: 24
-----
<DeviceName, B, T, L>
< \\.\PhysicalDrive11, 0, 0, 20>

<FcId, WWN, PortWWN, L>
<x010300, 50:0a:09:80:88:7d:2e:52, 50:0a:09:82:88:7d:2e:52, 5120>

<DeviceName, B, T, L>
< \\.\PhysicalDrive13, 0, 0, 22>

<FcId, WWN, PortWWN, L>
<x010300, 50:0a:09:80:88:7d:2e:52, 50:0a:09:82:88:7d:2e:52, 5632>

<DeviceName, B, T, L>
< \\.\PhysicalDrive3, 0, 1, 0>

<FcId, WWN, PortWWN, L>
  
```

An updated driver can be loaded by using the AutoPilot installation process, which is described in the Emulex driver download. AutoPilot installs the driver and the HBAAnywhere application without needing human interaction. The HBAAnywhere tool is required to update the firmware on the HBA.

The Emulex driver and utility-combined driver are available at the [Avago Technologies Support Document and Download](#) page.

If the Emulex driver version is incorrect, see the section titled “Performing an Unattended Installation” in the “Emulex Drivers for Windows User Manual” to install the Emulex driver and utility drivers.

To download the new firmware to the HBA (example shown in Figure 8), run the following command:

```
C:\Program Files (x86)\Emulex\Utils\HBAnywhere\hbcamd<wnn filename>.
```

Figure 8) HBA firmware update (example).

```
PS C:\Program Files (x86)\Emulex\Util\HBAnywhere> .\hbcamd download 10:00:00:00:c9:72:46:3f zf282a4.all
Downloading zf282a4.all to hba 10:00:00:00:c9:72:46:3f
```

Note: This step allows you to download specific firmware to a specific card.

Directory Structure

The following directory structure is expected in the X: \SANKIT directory.

```
Directory of E:\
<DIR>          E:\NetApp\DSM
<FILE>        E:\NetApp\DSM\ntap_win_mpio_3.4_setup_x64.msi
<DIR>          E:\NetApp\Emulex\Bios
<FILE>        E:\NetApp\Emulex\Bios\mb212a6.zip
<FILE>        E:\NetApp\Emulex\Bios\mf192a1.zip
<FILE>        E:\NetApp\Emulex\Bios\ob212a6.zip
<FILE>        E:\NetApp\Emulex\Bios\wb212a6.zip
<FILE>        E:\NetApp\Emulex\Bios\we412a7_etc.zip
<FILE>        E:\NetApp\Emulex\Bios\wf282a4.zip
<FILE>        E:\NetApp\Emulex\Bios\ym282a4.all
<FILE>        E:\NetApp\Emulex\Bios\ym282a4.zip
<DIR>          E:\NetApp\Emulex\Driver
<FILE>        E:\NetApp\Emulex\Driver\elxocm-windows-5.00.52.03-2.exe
<FILE>        E:\NetApp\Emulex\Driver\elxocm-windows-x64-5.01.09.04-1.exe
<FILE>        E:\NetApp\Emulex\Driver\HbaBOE.dll
<FILE>        E:\NetApp\Emulex\Driver\HbaCmd.exe
<FILE>        E:\NetApp\Emulex\Driver\RmApi.dll
<FILE>        E:\NetApp\Emulex\Driver\rmapijni.dll
<FILE>        E:\NetApp\Emulex\Driver\zf282a4.all
<DIR>          E:\NetApp\FCInfo
<FILE>        E:\NetApp\FCInfo\fcinfo_amd64.msi
<DIR>          E:\NetApp\Logs
<DIR>          E:\NetApp\Scripts
<FILE>        E:\NetApp\Scripts\SANKit1.0.ps1
<DIR>          E:\NetApp\WHUK
<FILE>        E:\NetApp\WHUK\netapp_windows_host_utilities_5.3_x64.msi
```

Code Base

This section provides the complete code base for SANKit using NetApp PowerShell Toolkit.

```
#####
# SANKit 1.0 Release
# Written by Chris Lionetti and B Sadhana
# (C) Jan 2011 NetApp
#
# This Powershell script is intended to install the following applications
# Emulex FC HBA Driver/BIOS
# Emulex FCoE HBA Driver/BIOS
# Microsoft FCInfo Tool (MSDN)
# Netapp Windows Host Utilities
# Netapp MPIO DSM Module
```

```

#
# This Powershell script is intended to run on the following OS
#   Microsoft Windows 2008r2
#   Microsoft Windows 2008r2sp1
#
# This Powershell script will store a text version of the LOG in the C:\SANKIT\LOG directory
# Only Available Options during Runtime are
#   -whatif          This option will show you what actions need to be taken. Will change
#                   NOTHING
#   -force           This option will ignore results of the three saftey checks.
#   -Verbose         This option will put all informational messages into the
#                   Event log/log/screen
#                   Normally Info Type messages are Supressed
#
#   Safety Checks
#   1). The script will abort if it finds EMC PowerPath, Navi CLI or Navi Agent installed
#   2). The script will not update drivers/BIOS on HBAs with LIVE LUNs attached
#   3). The script will not install NetApp DSM or WHUK if a NetApp WWN is not
#       discovered on SAN
#
#   Valid Exit Codes
#   0      SANKit Successfully completed.
#   1      SANKit Aborted due to a fault or missing pre-requisite such as No HBAs detected
#   2      SANKit Completed, but needing a reboot that it could not do. Please reboot and
#         rerun
#####
param( [Switch]$whatif,
       [switch]$force,
       [switch]$verbose
       )
$RebootRequired = $SuppressReboot = $False

function PostEvent([String]$TextField, [string]$EventType)
{ # Subroutine to Post Events to Log/Screen/EventLog
  $outfile = "C:\sankit\logs\netapp.log"
  if (!(test-path $OUTFILE)) { $suppress = mkdir c:\SANKit\Logs }
  if (!(test-path HKLM:\SYSTEM\CurrentControlSet\Services\Eventlog\application\SanKit) )
  { New-Eventlog -LogName Application -source SANKit
    PostEvent "Creating Eventlog\Application\SANKit Eventlog Source" "Warning"
  }
  else
  { switch -wildcard ($Eventtype)
    { "Info*" { $color="gray"
      "Warn*" { $color="green"
      "Err*" { $color="yellow"
      "Cri*" { $color="red"
              $EventType="Error"
      }
      Default { $color="gray"
    }
  }
  if (!(($verbose) -and ($EventType -eq "Information")))
  { write-host "- $textfield -foregroundcolor $color
    write-Eventlog -LogName Application -Source SANKit -EventID 1 -Message ` #continued ->
      $TextField -EntryType $EventType -Computername "." -category 0
      $textfield | out-file -filepath $outfile -append
  }
  }
}

function FindApp([String]$AppName) # Subroutine to Look for a Specific Application Installation
{ $TestResult=[boolean](get-itemproperty ` #continued ->
  'HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall\*' ` #continued ->
  | where { $_.displayname -like $AppName} )
  if ($TestResult) { PostEvent "Found the $AppName Tool installed" "Warning" }
  Return $TestResult
}

function WaitForInstallComplete([String]$AppName) # Loop until Install completes, or err in 30S
{ $TestResult=$False
  $StartCount=1
  While ($StartCount -lt 5)
  { Write-host "- Waiting until Installation Completes"
    start-sleep(15)
    $installedApps=get-itemproperty 'HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall\*'

```

```

foreach ($app in $installedApps)
{
    if ($app.displayname -like $AppName)
    {
        PostEvent "Detected $AppName Installed Correctly" "warning"
        $TestResult=$True
        $StartCount = 5
    }
}
$StartCount=$StartCount+1
}
if ($TestResult)
{
    Return
}
Else
{
    $ErrText=$Appname+"FAILED Installation. Timed Out during installation"
    PostEvent $ErrText "Critical"
    Exit 1 # Exit Code for Failure of SANKIT to Install
}
}

function PrintStartBanner # SANKit Banner
{
    cls
    Write-host "-----"
    Write-host "- SANKit 1.0 NetApp Starting Execution -"
    Write-host "-----"
    if ($verbose)
    {
        Write-host "- All Relevent Logs posted to NT Event Viewer - Source = SANKIT"
        Write-host "- All White Text is Screen Output Only for interactive users"
        Write-host "- All Gray Text is Informational (suppressed unless -verbose is used)"
        Write-host "- All Green Text is Successfully Action (Warning)"
        Write-host "- All Yellow Text is Incomplete Action (Error)"
        Write-host "- All Red Text is Failure - Abort Action (Critical)"
    }
    $EventText="Post Event SANKit start "+"`r`n"+ ` #continued next line ->
    "Runtime Switches; Whatif =$whatif, The Force =$force, Verbose = $verbose "
    PostEvent $EventText "Warning"
    $EventText= "Supported HBAs;`r`nEmulex LP1050 - Driver 7.2.32.002 Bios 2.82a4"+"`r`n"+`
    "Emulex LP10000 - Driver 7.2.32.002 Bios 1.92a1"+"`r`n"+`
    "Emulex LP111 - Driver 7.2.32.002 Bios 2.82a4"+"`r`n"+`
    "Emulex LPe111 - Driver 7.2.32.002 Bios 2.82a4"+"`r`n"+`
    "Emulex LP1150 - Driver 7.2.32.002 Bios 2.82a4"+"`r`n"+`
    "Emulex LPe1150 - Driver 7.2.32.002 Bios 2.82a4"+"`r`n"+`
    "Emulex LP11002 - Driver 7.2.32.002 Bios 2.82a4"+"`r`n"+`
    "Emulex LPe1205 - Driver 7.2.32.002 Bios 1.11a5"+"`r`n"+`
    "Emulex LP1250 - Driver 7.2.32.002 Bios 2.82a4"+"`r`n"+`
    "Emulex LP12000 - Driver 7.2.32.002 Bios 2.82a4"+"`r`n"+`
    "Emulex LP12002 - Driver 7.2.32.002 Bios 2.82a4"+"`r`n"+`
    "Emulex OCel10102-F Driver 7.2.32.002 Bios 2.102.200.17"
    PostEvent $EventText "Information"
    PostEvent "Supported Version of Windows Host Utility Kit = 5.3" "information"
    PostEvent "Supported Version of NetApp MPIO DSM = 3.4.1" "information"
}

# MAIN CODE STARTS HERE. Everything above is Functions
PrintStartBanner
# Copy SANKIT code to the C:\SANKIT\Netapp Directory
if (test-path "C:\SANKit\NetApp")
{
    PostEvent "SANKIT Detected at C:\SANKIT\NETAPP. No Need to Copy" "Warning"
}
else
{
    if ($whatif)
    {
        PostEvent "WHATIF : Copying SANKit from X:\SANKIT\NETAPP to Local C:\SANKIT\NETAPP" "Warning"
    }
    Else
    {
        PostEvent "Copying SANKit from X:\SANKIT\NETAPP to Local C:\SANKIT\NETAPP" "Warning"
        if (test-path "X:\SANKIT\NetApp")
        {
            xcopy x:\sankit\* c:\sankit\ /s /e /c /q /g /h /r /k /y
        }
    }
    else
    {
        PostEvent "Copy SANKIT to C:\SANKIT\NETAPP and run or mount X such that `#continued ->
        X:\SANKit\NetApp exists and re-run" "Critical"
        Exit 1 # Exit Code for Failure of SANKIT to Install
    }
}

```



```

"1250*" { $HBADrv = "7.2.32.002"; $HBAFw = "2.82A4"; $FWDriver = "ob282a4.all" }
Default { $HBADrv = $HBAFw = $FWDriver = "" }
}
if (!$EmulexSupportedCardPresent) { #if value true, dont want flipping back if later card false
{ $EmulexSupportedCardPresent = [Boolean](!$FWDriver -eq "") }
}
if (!$FWDriver -eq "")
{ $text= "Found HBA"+$HBA.Manufacturer+ " : "+$HBA.Model+", Driver " `#Continued next line
+$HBA.DriverVersion+", Firmware "+$HBA.FirmwareVersion+", Serial"+$HBA.SerialNumber
PostEvent $text "warning"
if ($HBA.DriverVersion -ne $hbadrv)
{ $text = "HBA WWN"+$HBA.Serial+$HBA.SerialNumber+" Running Driver "+`#continued next line
$HBA.DriverVersion+" Will be updated to Driver Version "+$HBADRV
PostEvent $Text "Error"
if ($LunsOnThisHBA -and !($Force))
{ PostEvent "LIVE LUNs Detected on this HBA. Cannot Update Driver" "Critical"
}
}
Else
{ if ($WhatIF)
{ PostEvent "WHATIF - Updating Emulex Driver on the HBA" "Warning"
}
else
{ PostEvent "Updating Emulex Driver on the HBA" "Warning"
$INSTALLCOMMAND = [System.Diagnostics.Process]::Start("cmd.exe", "`#continued
/c start /wait C:\SANKit\NetApp\Emulex\Driver\elxocm-windows-5.00.52.03-2 /q")
Invoke-expression $INSTALLCOMMAND | out-null
$INSTALLCOMMAND.WaitForExit()
$RebootRequired=$True #Driver Update requires a Reboot
}
}
}
if ($HBA.FirmwareVersion -ne $hbafw)
{ $text = "HBA Serial #"+$HBA.SerialNumber+" Firmware "+$HBA.FirmwareVersion+" `#continued
Will update to Firmware "+$HBAFw
PostEvent $Text "Error"
if ($LunsOnThisHBA)
{ PostEvent "LIVE LUNs Detected on this HBA. Cannot Update Firmware" "Critical"
}
}
Else
{ $INSTALLCOMMAND = [System.Diagnostics.Process]::Start("cmd.exe", "`#continued next line
/c start /wait C:\SANKit\NetApp\Emulex\Driver\elxocm-windows-5.00.52.03-2 /q")
Invoke-expression $INSTALLCOMMAND | out-null
$INSTALLCOMMAND.WaitForExit()
PostEvent "Sleeping for 60 seconds to let the fwupgrade complete" "Information"
Start-Sleep -s 60
$findWWN=C:\sankit\netapp\Emulex\driver\hbacmd.exe listhbas | Where{$_ -match "Port WWB"}
$theWWN=$findwwn[$ai].trimstart('Port WWN :')
$INSTALLCOMMAND = "c:\sankit\NetApp\Emulex\driver\Hbacmd.exe download "+ `#continued->
$theWWN+" c:\sankit\netapp\emulex\bios\"+$FWDriver
postevent $InstallCommand "Information"
Invoke-Expression $INSTALLCOMMAND
PostEvent "Sleeping for 60 seconds to let the fwupgrade complete" "Information"
Start-Sleep -s 60
$RebootRequired = $True
}
}
}
else
{ $text = "Non-Supported HBA Model "+$HBA.Model+" Running Driver "+ `#continued ->
$HBA.DriverVersion+" BIOS "+$HBA.FirmwareVersion
PostEvent $text "Error"
}
}
}
$ai=$ai+1
}
if (!$EmulexSupportedCardPresent)
{ PostEvent "Exiting SANKit! No Supported Emulex HBA found" "Critical"
exit 1 # Exit Code for Failure of SANKIT to Install
}
}
if (test-path C:\sankit\netapp\Emulex\driver\hbacmd.exe) # Find the NetApp WWNs.
{ $MissingHBACMD=$NetAppTargetPresent=$False

```

```

$findWWN=C:\sankit\netapp\emulex\driver\hbcamd.exe listhbas | where {$_ -match "Port WWN"}
foreach ($line in $findwwn)
{ $wn=$line.trimstart('Port WWN :')
  $Targetlist = C:\sankit\netapp\emulex\driver\hbcamd.exe allnodeinfo $wn | '#continued->
    where {$_ -match "50:0A:09"}
  if ($Targetlist)
  { $NetAppTargetPresent=[boolean]$Targetlist
    $text="Found Valid NetApp Target on Host WWN "+$wn
    postevent $text "Warning"
  }
  else
  { $EventText="No Valid NetApp Target on Host WWN "+$wn+" Check Cable and Zoning"
    postevent $EventText "Error"
  }
}
}
}
Else
{ $MissingHBACMD=$True
  PostEvent "WHATIF : HBACMD not present, cannot detect NetApp WWNs" "Error"
}
if ($NetAppTargetPresent -or $Force -or ($MissingHBACMD -and $WhatIf))
{ if (!(findapp "NetApp Windows Host U*))
  { $RebootRequired=$false
    if (!$whatif)
    { PostEvent "App named Netapp Windows Host U* Not Found. Starting Installation" "Error"
      $INSTALLCOMMAND = "msiexec /package '#continued->
        c:\SANKit\Netapp\WHUK\netapp_windows_host_utilities_5.3_x64.msi /log'#continued->
        C:\sankit\netapp\WHUK\WhukInstall.log /quiet /norestart MULTIPATHING=1"
      Invoke-expression $InstallCommand
      WaitForInstallComplete "NetApp Windows Host U*"
    }
    else
    { PostEvent "WHATIF : App Named NetApp Windows Host U* Not Found. Starting Install" "Error"
    }
  }
}
# Netapp DSM Installation Kit Loop
if (!(findApp "Data ONTAP DSM*"))
{ $RebootRequired=$True
  if (!$WhatIf)
  { PostEvent "App named $AppName Not Found. Starting Installation" "Error"
    $INSTALLCOMMAND = "msiexec /package c:\sankit\netapp\dsm\ntap_win_mpio_3.4_setup_x64.msi
      /quiet /log c:\sankit\netapp\logs\DSMlogfile.txt LICESNCECODE=VUDPMTRYAUDCMA
      USESYSTEMACCOUNT=1"
    Invoke-expression $InstallCommand
    WaitForInstallComplete "DATA ONTAP DSM*"
    $RebootRequired=$True
  }
  else
  { PostEvent "WHATIF : App named $AppName Not Found. Starting Installation" "Error"
  }
}
}
}
Else
{ PostEvent "No NetApp Target Discovered, Skipping DSM and WHUK Install" "Error"
}
if ($SuppressReboot -and $RebootRequired -and !($force))
{ PostEvent "Reboot required to complete, but LUNs online, Reboot been suppressed" "Error"
  exit 2 # Exit Code 2 means REBOOT REQUIRED, but otherwise success
}
if ($rebootrequired)
{ PostEvent "Force reboot. Rerun the SANKit after reboot,ensure all installed correctly" "Error"
  $INSTALLCOMMAND = "shutdown -r -t 90"
  if (!$whatif)
  { Invoke-Expression $INSTALLCOMMAND
  }
  Else
  { PostEvent "WHATIF : Intercepting Reboot Command - Not Rebooting" "Error"
  }
}
}
PostEvent "SANKit is Ending - Complete" "Warning"
exit 0 # Exit code Success

```

Appendix: PSTK Module Release Notes

PSTK unifies all of the NetApp PowerShell modules into a single package, with support for NetApp FAS and E-Series storage systems and EF-Series all-flash arrays. A unified toolkit provides end-to-end automation and storage management across NetApp storage platforms.

Data ONTAP PSTK

The Data ONTAP PowerShell module contains over 1,800 cmdlets, enabling the administration of NetApp FAS storage systems using the NetApp Manageability SDK. Full cmdlet sets are available for both 7-Mode and clustered Data ONTAP. PSTK also contains several cmdlets aimed at storage administration on Windows hosts, including:

- Creating virtual disks
- Resizing virtual disks
- Reclaiming space in virtual disks
- Copying files
- Deleting files
- Reclaiming space on host volumes

SANtricity PSTK

The SANtricity PowerShell module contains over 200 cmdlets, enabling the storage administration of NetApp E-Series storage systems and EF-Series all-flash arrays. The SANtricity PowerShell module leverages the NetApp SANtricity web services proxy to provide a distributed management solution. Use cases range from simple volume or disk monitoring to complex environment setup and teardown. By providing a rich object model, the PSTK provides the PowerShell scripter with valuable insight into the storage objects. Major features include:

- Discovering NetApp E-Series and EF-Series storage arrays
- Creating and deleting volume groups and pools
- Creating and deleting volumes
- Creating and deleting mirror groups, Snapshot copies, consistency groups, and others
- Configuring hosts
- Monitoring health and performance
- Performing controller firmware and NVSRAM upgrades
- Collecting hardware inventory and event logs

System Requirements

Table 40) General system requirements.

General	Version
Microsoft Windows PowerShell	3.0 and later
Microsoft .NET Framework	4.5.2 and later
Microsoft Windows	2003, 2008, 2012, 7, 8, and later

Table 41) Data ONTAP PSTK requirements.

Data ONTAP PSTK	Version
Data ONTAP storage controllers: 7-Mode and clustered Data ONTAP	7.3.x and later

Table 42) SANtricity PSTK requirements.

SANtricity PSTK	Version
E-Series and EF-Series SANtricity management plug-ins for web services (download here)	1.3 and later

Installation Requirements

Make sure that the following PSTK installation requirements have been met:

- Installer (preferred): Download and run the installer package.
- Zip (advanced): Download the zipped file and extract it to the PSMModule directory.

Support and Additional Information

PSTK offers community support. Create a new discussion in the following community sites:

- [Data ONTAP related discussions](#)
- [SANtricity, E-Series, and EF-Series related discussions](#)

Recommended Additional Reading

The following books are recommended for building your PowerShell skills:

- Don Jones, Jeffery D. Hicks, "Learn Windows PowerShell in a Month of Lunches" (Manning, 2012), 368 pages
- Don Jones, Jeffery D. Hicks, "Learn PowerShell Toolmaking in a Month of Lunches" (Manning, 2012), 312 pages

Version History

Version	Date	Author	Document Version History
Version 1.0	November 2015	Chris Lionetti	Initial release
Version 1.1	February 2016	Shashanka SR	Added SANtricity cmdlets (PSTK 4.1)
Version 1.2	October 2016	Shashanka SR	PSTK 4.3 update

Refer to the [Interoperability Matrix Tool \(IMT\)](#) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

Copyright Information

Copyright © 1994–2016 NetApp, Inc. All rights reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

Trademark Information

NetApp, the NetApp logo, Go Further, Faster, AltaVault, ASUP, AutoSupport, Campaign Express, Cloud ONTAP, Clustered Data ONTAP, Customer Fitness, Data ONTAP, DataMotion, Flash Accel, Flash Cache, Flash Pool, FlashRay, FlexArray, FlexCache, FlexClone, FlexPod, FlexScale, FlexShare, FlexVol, FPolicy, GetSuccessful, LockVault, Manage ONTAP, Mars, MetroCluster, MultiStore, NetApp Fitness, NetApp Insight, OnCommand, ONTAP, ONTAPI, RAID DP, RAID-TEC, SANshare, SANtricity, SecureShare, Simplicity, Simulate ONTAP, SnapCenter, SnapCopy, Snap Creator, SnapDrive, SnapIntegrator, SnapLock, SnapManager, SnapMirror, SnapMover, SnapProtect, SnapRestore, Snapshot, SnapValidator, SnapVault, SolidFire, StorageGRID, Tech OnTap, Unbound Cloud, WAFL, and other names are trademarks or registered trademarks of NetApp, Inc. in the United States and/or other countries. All other brands or products are trademarks or registered trademarks of their respective holders and should be treated as such. A current list of NetApp trademarks is available on the web at <http://www.netapp.com/us/legal/netapptmlist.aspx>. TR-4475-1016