



# NetApp SolidFire PowerShell Tools User Guide

Version: 1.5.1

April 2018

# Copyright Information

Copyright © 1994-2018 NetApp, Inc. All Rights Reserved.

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 document 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, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.

# Table of Contents

<b>Introduction</b>	<b>1</b>
<b>Supported OS and OS-level Virtualization</b>	<b>1</b>
<b>Installing or Upgrading SolidFire PowerShell Tools</b>	<b>2</b>
<b>How to Use SolidFire PowerShell Tools</b>	<b>3</b>
Listing Available Functions	3
Accessing Embedded Help	4
Parameter Sets	4
Managing Connections to a SolidFire Cluster	5
Connecting to a SolidFire Cluster	5
Connecting to a SolidFire Node	5
Disconnecting from a SolidFire Cluster or Node	6
Changing API Versions	6
Global Variables for All Functions	7
Common Parameters	8
Return Object Descriptions	9
Accessing Return Value Reference Documentation	9
Accessing Return Values Using Get-Help	9
Leveraging Get-Member to Inspect Return Objects	10
<b>Contacting NetApp Support for SolidFirePowerShell Tools</b>	<b>11</b>

# Introduction

SolidFire PowerShell Tools is a collection of Microsoft® Windows® PowerShell functions that use SolidFire API to control a SolidFire storage system. These functions allow administrators to query for information, make changes to objects in a storage system, and develop complex scripts on a single platform. You can use this module with other modules and snap-ins, such as VMware® PowerCLI and Cisco® UCS PowerTool, to extend capabilities throughout the infrastructure.

Any user with a SolidFire storage system and Windows PowerShell can take advantage of SolidFire PowerShell Tools. Before you use SolidFire PowerShell Tools, you should have an understanding of Windows PowerShell functions. The SolidFire PowerShell Tools module can be obtained through [PowerShell Gallery](#), the SolidFire Support [BrickFTP](#) site, or [GitHub](#).

## Supported OS and OS-level Virtualization

The following operating systems and container software are supported:

OS and Containers	Required PowerShell Version
Windows® Server 2016	PowerShell 6.0
MacOS® 10.13	PowerShell 6.0
CentOS™ 7.3	PowerShell 6.0
Ubuntu 16.04	PowerShell 6.0
Windows® Server 2016	PowerShell 5.1 and earlier
Microsoft® Windows® 10	PowerShell 5.1 and earlier
Windows® Server 2012 R2 64-bit	PowerShell 5.1 and earlier
Docker*	N/A

\*Runs a container with Ubuntu and the SolidFire PowerShell tools pre-installed.

# Installing or Upgrading SolidFire PowerShell Tools

You can install or upgrade to the latest NetApp SolidFire PowerShell Tools module according to these instructions.

## Prerequisites

- Administrative privileges to be able to complete the installation.
- 64-bit operating system needed to run the installer.

## Procedure

The SolidFire PowerShell Tools Module can also be installed directly on multiple systems by following these instructions:

- [Windows](#)
- [MacOS](#)
- [Linux](#)
- [Docker](#)

# How to Use SolidFire PowerShell Tools

The following topics describe ways to access available functions for SolidFire PowerShell Tools, manage connections to a SolidFire node, and find additional cmdlet parameter and return object information.

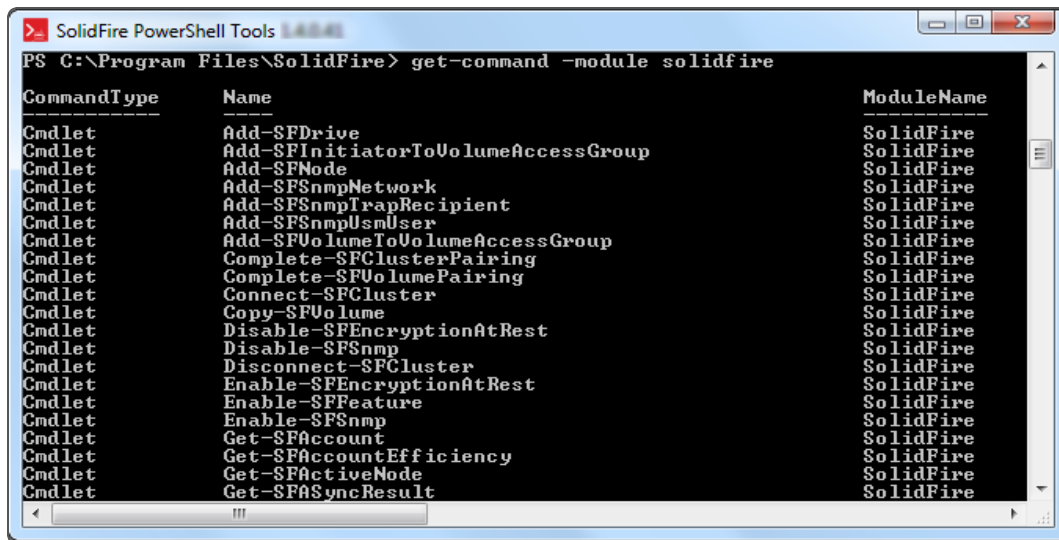
## Listing Available Functions

The available functions for SolidFire PowerShell Tools can be explored using the native Get-Command PowerShell Tools cmdlet.

### Procedure

1. In the command line interface, type `Get-Command -Module SolidFire` (for PowerShell 6.0 and later on Mac and Linux, type `Get-Command -Module SolidFire.Core`).

The list of available commands appears.



```
PS C:\Program Files\SolidFire> get-command -module solidfire
```

CommandType	Name	ModuleName
Cmdlet	Add-SFDrive	SolidFire
Cmdlet	Add-SFInitiatorToVolumeAccessGroup	SolidFire
Cmdlet	Add-SFNode	SolidFire
Cmdlet	Add-SFSnmpNetwork	SolidFire
Cmdlet	Add-SFSnmpTrapRecipient	SolidFire
Cmdlet	Add-SFSnmpUser	SolidFire
Cmdlet	Add-SFVolumeToVolumeAccessGroup	SolidFire
Cmdlet	Complete-SFClusterPairing	SolidFire
Cmdlet	Complete-SFVolumePairing	SolidFire
Cmdlet	Connect-SFCluster	SolidFire
Cmdlet	Copy-SFVolume	SolidFire
Cmdlet	Disable-SFEncryptionAtRest	SolidFire
Cmdlet	Disable-SFSnmp	SolidFire
Cmdlet	Disconnect-SFCluster	SolidFire
Cmdlet	Enable-SFEncryptionAtRest	SolidFire
Cmdlet	Enable-SFFeature	SolidFire
Cmdlet	Enable-SFSnmp	SolidFire
Cmdlet	Get-SFAccount	SolidFire
Cmdlet	Get-SFAccountEfficiency	SolidFire
Cmdlet	Get-SFActiveNode	SolidFire
Cmdlet	Get-SFASyncResult	SolidFire

2. Type a search term with an asterisk before and after the term to filter the command list:  
`Get-Command *volumepair* -Module SolidFire`  
 (for PowerShell 6.0 and later on Mac and Linux, type `Get-Command *volumepair* -Module SolidFire.Core`).

The filtered list of available commands appears.

```

SolidFire PowerShell Tools
Cmdlet      Remove-SFInitiatorFromVolumeAccessGroup  SolidFire
Cmdlet      Remove-SFVolume                          SolidFire
Cmdlet      Remove-SFVolumeAccessGroup               SolidFire
Cmdlet      Remove-SFVolumeFromVolumeAccessGroup     SolidFire
Cmdlet      Remove-SFVolumePair                      SolidFire
Cmdlet      Set-SFVolume                             SolidFire
Cmdlet      Set-SFVolumeAccessGroup                  SolidFire
Cmdlet      Set-SFVolumePair                         SolidFire
Cmdlet      Start-SFVolumeBackup                     SolidFire
Cmdlet      Start-SFVolumePairing                    SolidFire
Cmdlet      Start-SFVolumeRestore                     SolidFire

PS C:\Program Files\SolidFire> get-command *volumepair* -module solidfire

CommandType      Name                                     ModuleName
-----
Cmdlet            Complete-SFVolumePairing                SolidFire
Cmdlet            Get-SFVolumePair                        SolidFire
Cmdlet            Remove-SFVolumePair                     SolidFire
Cmdlet            Set-SFVolumePair                        SolidFire
Cmdlet            Start-SFVolumePairing                    SolidFire

PS C:\Program Files\SolidFire>

```

## Accessing Embedded Help

SolidFire PowerShell Tools contains help examples that are accessible through the command line. Help content includes details about each command and examples of each function in use.

### Procedure

1. In the command line interface, type `Get-Help <cmdlet>`.

The cmdlet description from embedded help appears.

```

SolidFire PowerShell Tools
NAME
    Connect-SFCluster

SYNOPSIS
    [Node/Cluster] Initiates a connection sequence that establishes a SolidFire

SYNTAX
    Connect-SFCluster [-Target] <string> [-Credential] <pscredential> [-Node] [-
    Connect-SFCluster [-Target] <string> [-Username] <string> [-Password] <string>

DESCRIPTION
    [Node/Cluster] Initiates a connection sequence that establishes a SolidFire

RELATED LINKS

REMARKS
    To see the examples, type: "get-help Connect-SFCluster -examples".
    For more information, type: "get-help Connect-SFCluster -detailed".
    For technical information, type: "get-help Connect-SFCluster -full".

```

**NOTE:** To view full cmdlet help, see [Accessing Return Values Using Get-Help](#).

## Parameter Sets

Many of the functions for SolidFire PowerShell Tools have parameter sets to allow multiple use cases. For example, parameter sets are used with the creation and modification of SolidFire objects, such as Accounts, Volumes, and Volume Access Groups.

You can identify parameter sets by using `Get-Help` for the function and reviewing the content under the Syntax section.

## Managing Connections to a SolidFire Cluster

All of the functions in SolidFire PowerShell Tools make direct calls to the SolidFire API. In order to manage authentication efficiently, a connection function has been developed for collecting target and authentication information.

### Connecting to a SolidFire Cluster

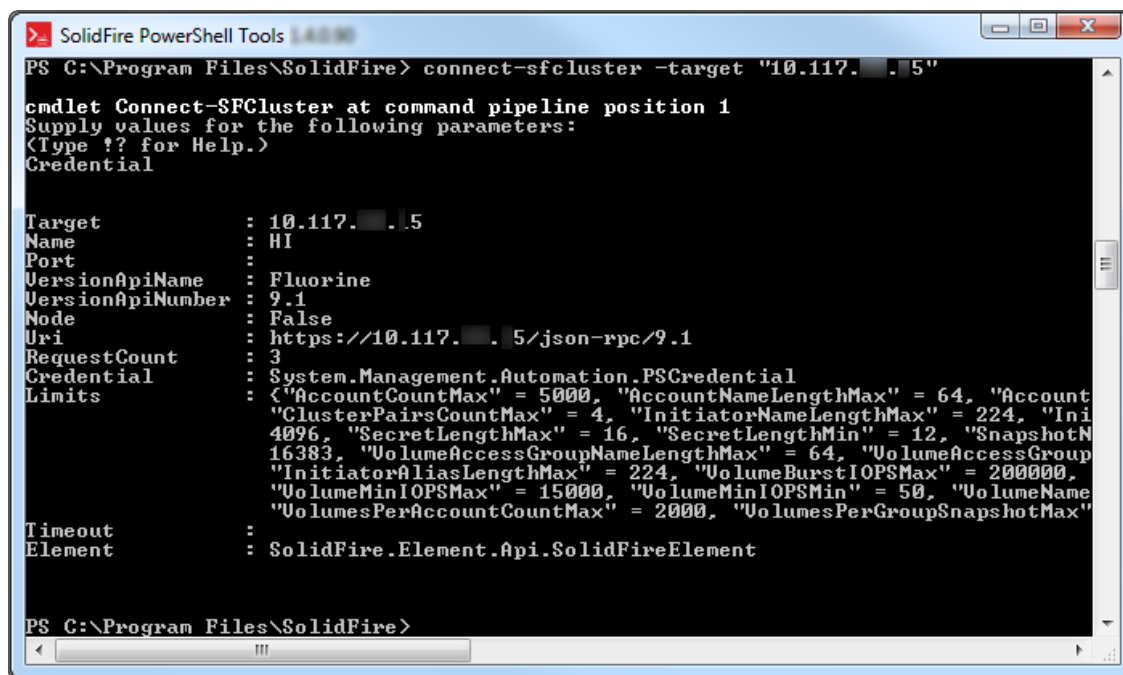
Use `Connect-SFCluster` to connect to a SolidFire cluster. The function collects SolidFire connection information, including target and authentication information from the user. `Connect-SFCluster` also supports connections to multiple SolidFire clusters.

By default, the `Connect-SFCluster` cmdlet queries the target cluster and sets the connection information to the latest API version on the cluster. This could also change the URI property to the version of Element OS you are using. See [Changing API Versions](#) to specify an API.

#### Procedure

1. In the command line interface, type `Connect-SFCluster -Target "<address>"`.

The following example shows a successful connection.



```
PS C:\Program Files\SolidFire> connect-sfcluster -target "10.117.1.5"

cmdlet Connect-SFCluster at command pipeline position 1
Supply values for the following parameters:
(Type ?? for Help.)
Credential

Target           : 10.117.1.5
Name             : H1
Port            :
VersionApiName   : Fluorine
VersionApiNumber : 9.1
Node            : False
Uri             : https://10.117.1.5/json-rpc/9.1
RequestCount     : 3
Credential       : System.Management.Automation.PSCredential
Limits           : {"AccountCountMax" = 5000, "AccountNameLengthMax" = 64, "Account
                  "ClusterPairsCountMax" = 4, "InitiatorNameLengthMax" = 224, "Ini
                  4096, "SecretLengthMax" = 16, "SecretLengthMin" = 12, "SnapshotN
                  16383, "VolumeAccessGroupNameLengthMax" = 64, "VolumeAccessGroup
                  "InitiatorAliasLengthMax" = 224, "VolumeBurstIOPSMax" = 200000,
                  "VolumeMinIOPSMax" = 15000, "VolumeMinIOPSMin" = 50, "VolumeName
                  "VolumesPerAccountCountMax" = 2000, "VolumesPerGroupSnapshotMax"
Timeout         :
Element         : SolidFire.Element.Api.SolidFireElement

PS C:\Program Files\SolidFire>
```

**NOTE:** If your connection to a SolidFire cluster is successful, the function `Connect-SFCluster` stores credentials and target information into a global variable `$SFConnection`. Multiple connections are also supported, and each successful connection is stored in the global array variable `$SFconnections`. See [Global Variables for All Functions](#).

### Connecting to a SolidFire Node

Use the `Connect-SFCluster` function with a `-Node` switch parameter to connect to a specific SolidFire node.

#### Procedure

1. In the command line interface, type `Get-SFNode | Select Name, ManagementIP, NodeID` to get the IP address for



the node.

2. Include `-Node` and provide the node IP address in order to connect.

```
Connect-SFCluster -Target <NodeIP> -UserName <AdminAccount> -Node:
```

## Disconnecting from a SolidFire Cluster or Node

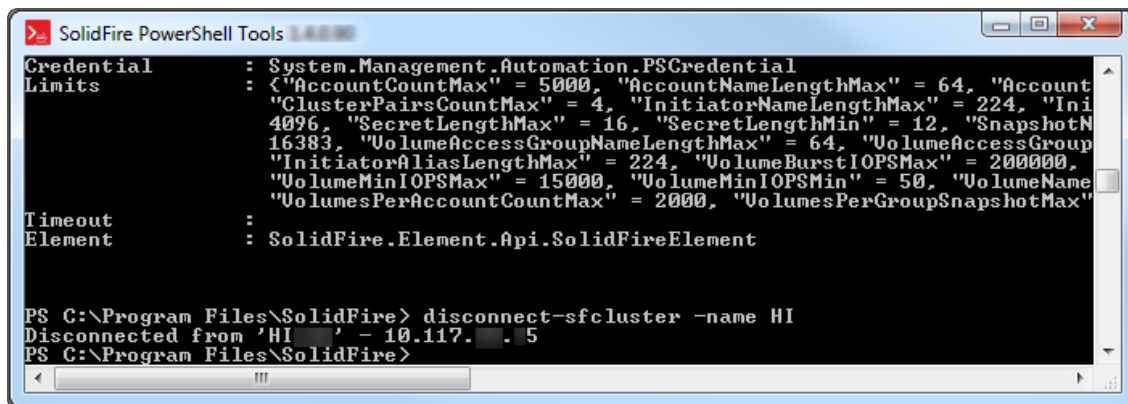
Use the `Disconnect-SFCluster` function to disconnect from a SolidFire cluster. The function also clears the `$SFConnection` and `$SFConnections` global variables from the session. This makes it easier to secure the shell if you wish to keep it active or work with a different SolidFire cluster.

You can disconnect a specific connection using the name of the connection from the `$SFConnection` or `$SFConnections` global variables. This name is either the cluster or the node name. See [Global Variables for All Functions](#) for an example.

### Procedure

1. In the command line interface, type `Disconnect-SFCluster` to disconnect from the cluster. You can add an optional extension `-Name <node or cluster name>` or optional `-Target` parameter to specify the IP address instead of the name.

The following example shows a successful disconnection.



## Changing API Versions

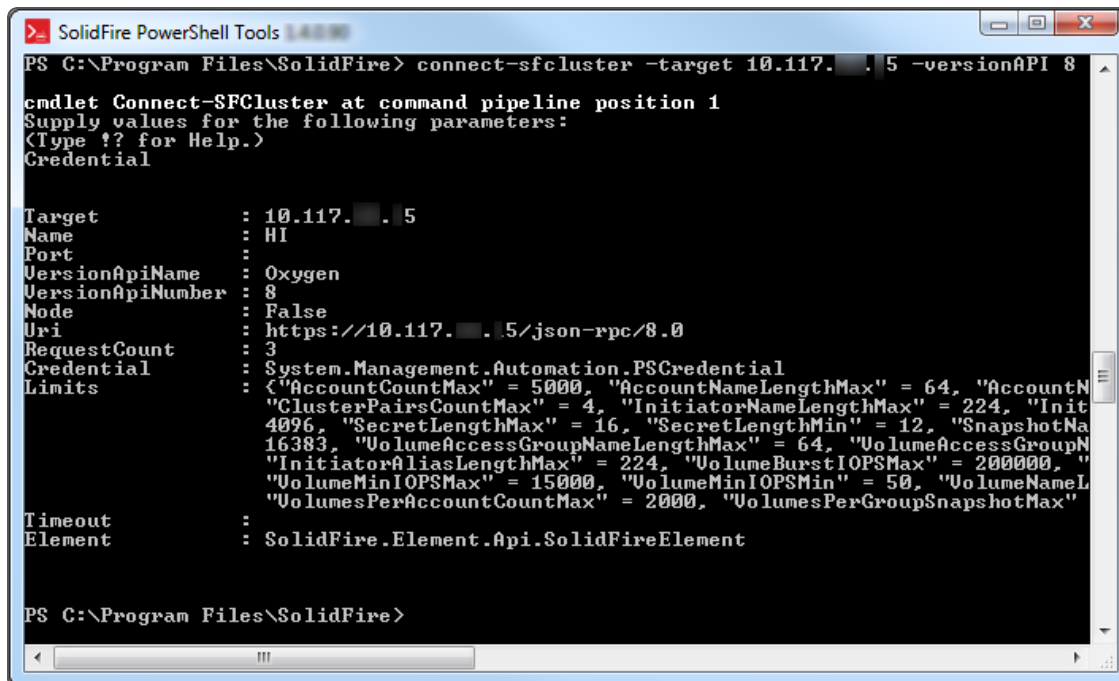
Use `-VersionApi` to specify a SolidFire API version. By default, the SolidFire PowerShell Tools module will use the most recent version of the SolidFire API available.

**NOTE:** Changing versions might produce unexpected results based on availability of features and possible API method changes between releases. Even if a connection to an API version works, not all new features might be available in the SolidFire PowerShell Tools module version that you have installed.

### Procedure

1. Type `Connect-SFCluster -Target <address> -VersionApi <version number>`.

The following example demonstrates connecting to a SolidFire cluster with API version 8.0 (Oxygen).



```

PS C:\Program Files\SolidFire> connect-sfcluster -target 10.117.1.5 -versionAPI 8

cmdlet Connect-SFCluster at command pipeline position 1
Supply values for the following parameters:
(Type '!' for Help.)
Credential

Target          : 10.117.1.5
Name            : HI
Port           : 
VersionApiName  : Oxygen
VersionApiNumber : 8
Node           : False
Uri            : https://10.117.1.5/json-rpc/8.0
RequestCount    : 3
Credential      : System.Management.Automation.PSCredential
Limits         : {"AccountCountMax" = 5000, "AccountNameLengthMax" = 64, "AccountNameLengthMin" = 12, "ClusterPairsCountMax" = 4, "InitiatorNameLengthMax" = 224, "InitiatorNameLengthMin" = 12, "SnapshotNameLengthMax" = 64, "SnapshotNameLengthMin" = 12, "VolumeAccessGroupNameLengthMax" = 64, "VolumeAccessGroupNameLengthMin" = 12, "VolumeBurstIOPSMax" = 200000, "VolumeBurstIOPSMIn" = 50, "VolumeMinIOPSMax" = 15000, "VolumeMinIOPSMIn" = 50, "VolumeNameLengthMax" = 64, "VolumeNameLengthMin" = 12, "VolumesPerAccountCountMax" = 2000, "VolumesPerGroupSnapshotMax" = 1000}
Timeout        : 
Element        : SolidFire.Element.Api.SolidFireElement

PS C:\Program Files\SolidFire>

```

## Global Variables for All Functions

If your connection to a SolidFire cluster is successful, the function `Connect-SFCluster` stores credentials and target information in a global variable `$SFConnection`. The information in this variable is used in API calls of other SolidFire PowerShell Tools functions. Multiple connections are also supported, and each successful connection is stored in the global array variable `$SFConnections`.

```

PS C:\Program Files\SolidFire> $sfconnection

Target      : 10.117.1.5
Name        : H1
Port        :
VersionApiName : Fluorine
VersionApiNumber : 9.1
Node        : False
Uri         : https://10.117.1.5/json-rpc/9.1
RequestCount : 3
Credential   : System.Management.Automation.PSCredential
Limits      : {"AccountCountMax" = 5000, "AccountNameLengthMax" = 64, "AccountNameLengthMin" = 12, "SnapshotNameLengthMax" = 64, "SnapshotNameLengthMin" = 12, "InitiatorNameLengthMax" = 224, "InitiatorNameLengthMin" = 12, "InitiatorAliasLengthMax" = 224, "InitiatorAliasLengthMin" = 12, "VolumeAccessGroupNameLengthMax" = 64, "VolumeAccessGroupNameLengthMin" = 12, "VolumeBurstIOPSMax" = 200000, "VolumeBurstIOPSMin" = 50, "VolumeMinIOPSMax" = 15000, "VolumeMinIOPSMin" = 50, "VolumeNameLengthMax" = 32, "VolumesPerAccountCountMax" = 2000, "VolumesPerGroupSnapshotMax" = 32}

Timeout     :
Element     : SolidFire.Element.Api.SolidFireElement

PS C:\Program Files\SolidFire> $sfconnections

Target      : 10.117.1.7
Name        : Auto
Port        :
VersionApiName : Fluorine
VersionApiNumber : 9.1
Node        : False
Uri         : https://10.117.1.7/json-rpc/9.1
RequestCount : 3
Credential   : System.Management.Automation.PSCredential
Limits      : {"AccountCountMax" = 5000, "AccountNameLengthMax" = 64, "AccountNameLengthMin" = 12, "SnapshotNameLengthMax" = 64, "SnapshotNameLengthMin" = 12, "InitiatorNameLengthMax" = 224, "InitiatorNameLengthMin" = 12, "InitiatorAliasLengthMax" = 224, "InitiatorAliasLengthMin" = 12, "VolumeAccessGroupNameLengthMax" = 64, "VolumeAccessGroupNameLengthMin" = 12, "VolumeBurstIOPSMax" = 200000, "VolumeBurstIOPSMin" = 50, "VolumeMinIOPSMax" = 15000, "VolumeMinIOPSMin" = 50, "VolumeNameLengthMax" = 32, "VolumesPerAccountCountMax" = 2000, "VolumesPerGroupSnapshotMax" = 32}

Timeout     :
Element     : SolidFire.Element.Api.SolidFireElement

Target      : 10.117.1.5
Name        : H1
Port        :
VersionApiName : Fluorine
VersionApiNumber : 9.1
Node        : False
Uri         : https://10.117.1.5/json-rpc/9.1
RequestCount : 3
Credential   : System.Management.Automation.PSCredential
Limits      : {"AccountCountMax" = 5000, "AccountNameLengthMax" = 64, "AccountNameLengthMin" = 12, "SnapshotNameLengthMax" = 64, "SnapshotNameLengthMin" = 12, "InitiatorNameLengthMax" = 224, "InitiatorNameLengthMin" = 12, "InitiatorAliasLengthMax" = 224, "InitiatorAliasLengthMin" = 12, "VolumeAccessGroupNameLengthMax" = 64, "VolumeAccessGroupNameLengthMin" = 12, "VolumeBurstIOPSMax" = 200000, "VolumeBurstIOPSMin" = 50, "VolumeMinIOPSMax" = 15000, "VolumeMinIOPSMin" = 50, "VolumeNameLengthMax" = 32, "VolumesPerAccountCountMax" = 2000, "VolumesPerGroupSnapshotMax" = 32}

Timeout     :
Element     : SolidFire.Element.Api.SolidFireElement

PS C:\Program Files\SolidFire>

```

## Common Parameters

All cmdlets, with the exception of `Connect-SFCluster` and `Disconnect-SFCluster`, have the common parameter `Target` and `SFConnection`. These common parameters are not in the Get-Help examples for each cmdlet but can be assumed to be present. The embedded Help within PowerShell Tools has the common parameter in its examples.

If the `-Target` parameter is included in the cmdlet, the cmdlet will run against all connections in `$SFConnections` whose name or target (IP address) matches using a wildcard pattern match. Results are written to output without indicating the target against which the cmdlet was run.

If the `-SFConnection` parameter is included in the cmdlet, the cmdlet will run against the specific `SFConnection` that was handed in through that parameter. You can inspect the `$SFConnections` session variable to find a specific `SFConnection` or you can pass the result of `Connect-SFCluster` into it.

Each cmdlet is configured to be run on either a cluster or node. Before processing the cmdlet against any target, the cmdlet will check the connection to make sure it matches the intended cluster or node. If there is no match, a non-terminating error message (as in the following example) appears that states it is skipping the command:

```
Get-SFNetworkConfig : Skipping command on connection 'Connection Name'. CmdLet requires Node connection.
```

All cmdlets will execute against all matching connections.

## Return Object Descriptions

Return values are fully documented as part of the .NET SDK documentation that is available online. There are three methods for inspecting cmdlet return values:

- [Accessing Return Value Reference Documentation](#)
- [Accessing Return Values Using Get-Help](#)
- [Leveraging Get-Member to Inspect Return Objects](#)

### Accessing Return Value Reference Documentation

Each return value is documented as part of the online documentation for the SolidFire .NET SDK. This documentation can be found on [GitHub](#).

### Accessing Return Values Using Get-Help

For any cmdlet included in SolidFire PowerShell Tools, type `Get-Help <cmdlet name> -Full` to return the following:

- A specific return type for the cmdlet that is described in the Outputs section.
- A URL to the related SolidFire .NET SDK reference page on GitHub.
- Examples of cmdlet use.

The following is an example of `Get-Help Get-SFVolume -Full`:

```

PS C:\Program Files\SolidFire> get-help get-sfvolume -full

NAME
    Get-SFVolume

SYNOPSIS
    [Cluster] Gets a list of volumes from the cluster.

SYNTAX
    Get-SFVolume [[-VolumeID] <long[]>] [[-ExcludeVUOLs]] [[-IncludeDeleted]] [-]
    Get-SFVolume [[-Name] <string[]>] [[-ExcludeVUOLs]] [[-IncludeDeleted]] [-]
    Get-SFVolume [-AccountID] <long[]> [[-ExcludeVUOLs]] [[-IncludeDeleted]] [-]
    Get-SFVolume [-Account] <Account[]> [[-ExcludeVUOLs]] [[-IncludeDeleted]] [-]

DESCRIPTION
    [Cluster] Gets a list of volumes from the cluster.

PARAMETERS
    -Account <Account[]>
        Specify account(s).

        Required?                true
        Position?                0
        Default value
        Accept pipeline input?    true (ByValue, ByPropertyName)
        Accept wildcard characters?

    -AccountID <long[]>
        Enter an Account ID or list of Account IDs.

        Required?                true
        Position?                0
        Default value
        Accept pipeline input?    true (ByValue, ByPropertyName)
        Accept wildcard characters?

    -ExcludeVUOLs

```

## Leveraging Get-Member to Inspect Return Objects

Use the built-in SolidFire PowerShell Tools `Get-Member` cmdlet to inspect return values.

### Procedure

1. In the command line interface, type `$<variable> = <Cmdlet with appropriate parameters>`
2. Type `$<variable> | Get-Member`.

The following example shows the result for `Get-SFVolume` with each return property listed.

```

PS C:\Program Files\SolidFire> $volumes = Get-SFVolume
PS C:\Program Files\SolidFire> $volumes | Get-Member

TypeName: SolidFire.Element.Api.Volume

Name           MemberType Definition
-----
Equals          Method    bool Equals(System.Object obj)
GetHashCode     Method    int GetHashCode()
GetType         Method    type GetType()
MkString        Method    string MkString()
ToString        Method    string ToString()
Access          Property  string Access {get;set;}
AccountID       Property  long AccountID {get;set;}
Attributes      Property  hashtable Attributes {get;set;}
BlockSize       Property  long BlockSize {get;set;}
CreateTime      Property  string CreateTime {get;set;}
DeleteTime      Property  string DeleteTime {get;set;}
Enable512e      Property  bool Enable512e {get;set;}
Iqn             Property  string Iqn {get;set;}
Name            Property  string Name {get;set;}
PurgeTime       Property  string PurgeTime {get;set;}
Qos             Property  SolidFire.Element.Api.QoSResult Qos {get;set;}
ScsiEUIDeviceID Property  string ScsiEUIDeviceID {get;set;}
ScsiNAADeviceID Property  string ScsiNAADeviceID {get;set;}
SliceCount      Property  long SliceCount {get;set;}
Status          Property  string Status {get;set;}
TotalSize       Property  long TotalSize {get;set;}
VirtualVolumeID Property  guid VirtualVolumeID {get;set;}
VolumeAccessGroups Property  long[] VolumeAccessGroups {get;set;}
VolumeID        Property  long VolumeID {get;set;}
VolumePairs     Property  SolidFire.Element.Api.VolumePair[] VolumePairs...

PS C:\Program Files\SolidFire>

```

3. If objects are more than one layer deep (see the QoS property in the example from the previous step), examine additional layers using the dot operator `$<variable.property> | Get-Member`.

## Contacting NetApp Support for SolidFirePowerShell Tools

If you have any questions or comments about this product, reach out to the development community at [ThePub](#). We also monitor the [GitHub PowerShell](#) repository for open issues or pull requests. Your feedback helps us focus our efforts on new features and capabilities.



1048 Pearl Street, Suite 250  
Boulder, Colorado 80302

Web: [netapp.com](http://netapp.com)  
Support: [mysupport.netapp.com](http://mysupport.netapp.com)

April 2018