

# 14

## Managing View with PowerCLI

VMware View provides 56 different PowerShell commandlets that you can use to configure, manage, and monitor the View environment. These commandlets are known as **View PowerCLI**, and enable View administrators to do everything from automating repetitive operations to using existing IT infrastructure management platforms to perform common View tasks. While not every aspect of the View environment can be managed or configured using View PowerCLI, many of the most common settings can.

In this chapter, we will learn:

- How to enable remote management on a View Connection Server so that you can use PowerShell remotely
- How to establish a remote PowerShell session
- What are the different View PowerCLI commandlets
- How to use each of the View PowerCLI commandlets

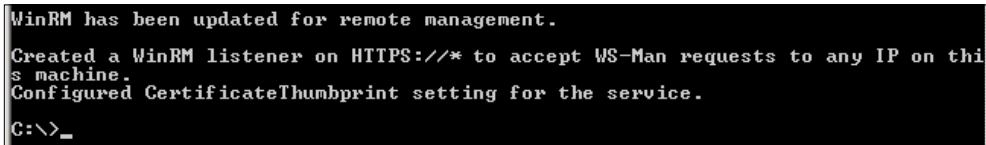
Using the information provided in this chapter, you should be able to reduce the time you spend in the View Manager Admin console by building scripts that can perform the actions more quickly.

### Enabling Windows Remote Management

Unlike vSphere PowerCLI, VMware View does not include a standalone installer to use to remotely manage View using PowerCLI. The View PowerCLI commands will only work when executed from a View Connection Server. To enable remote management, we must enable **Windows Remote Management**, also known as **WinRM**.

WinRM is based on the WS-Management Protocol, which is a SOAP-based protocol used to enable interoperability between hardware and operating systems from different vendors. We will use WinRM to establish remote PowerShell connections to a View Connection Server, which will enable us to run commands from that server without actually having to log in to the server console. The following steps outline how to enable WinRM:

1. Log in to the View Connection Server that you will use for your remote sessions.
2. Enable and start the **Windows Remote Management (WS-Management)** service. This service should be set to start automatically.
3. From a Windows command prompt on the server, execute the following command to enable inbound WinRM requests over HTTPS:  
`winrm quickconfig -transport:https`
4. When prompted, answer `y` to approve the operation, and verify that the operation succeeded, as shown in the following screenshot:



```
WinRM has been updated for remote management.
Created a WinRM listener on HTTPS://* to accept WS-Man requests to any IP on this machine.
Configured CertificateThumbprint setting for the service.
C:\>_
```

5. If Windows firewall is enabled on the View Connection Server, create a firewall rule that allows TCP port 5986 inbound. This is the port used when connecting to WinRM over SSL.

WinRM is now configured and is available to any users with local administrative access to the server.

## Establishing a remote View PowerCLI session

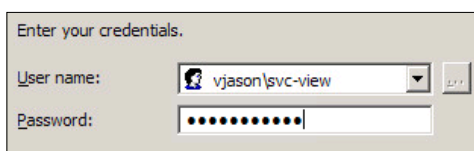
Once WinRM is enabled, you can connect to the View Connection Server remotely over a PowerShell session. The following steps outline how to establish a remote PowerShell session, and then enable the View PowerCLI commandlets:

1. Open a PowerShell window on the computer you will use to remotely manage VMware View.

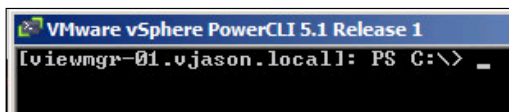
- Use the following command to initiate a remote PowerShell session. You will need to provide the FQDN of the View Connection Server, a user ID that has administrative access to both View and the View Connection Server, and the `-UseSSL` option:

```
Enter-PSSession -ComputerName VIEWMGR-01.vjason.local -UseSSL -  
Credential vjason\svc-view
```

- A **Windows PowerShell Credential Request** window will open as shown in the following screenshot; provide the password for the user account specified in the `-Credential` option of the previous step and click on **OK**:



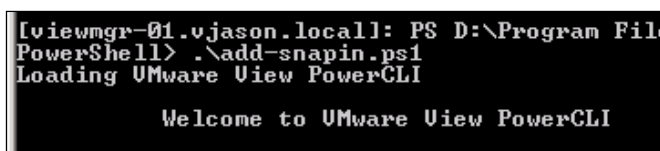
- The PowerShell window will now display a command prompt that includes the name of the View Connection Server as shown in the following screenshot. You are now running a PowerShell session on that server, from the local drive indicated in the console. You can change drives by selecting another drive letter exactly as you would if you were logged on directly using the console:



- Change into the `Program Files\VMware\VMware View\Server\extras\PowerShell` folder on the View Connection Server.
- Execute the following command to run the script that will load the View PowerCLI commandlets:

```
Add-snapin.ps1
```

Once the script finishes running, **Welcome to VMware View PowerCLI** should appear as shown in the following screenshot. You are now able to use PowerCLI to manage View:



## Sample data for this chapter

The following values will be used to create our sample View PowerCLI commands later in the chapter. Some portions of the text are highlighted; those represent objects that are created automatically within vCenter but are not visible to the end user. These objects must be included in the PowerCLI command or it will not work:

Object name	Value
AD domain name	vjason.local
AD group used for sample commands	Finance_View_Users
AD user used for sample commands	Jason Ventresco (vjason\jventresco)
Destination vCenter folder for View desktops, including path	/RTP/Desktops
Datstores used for desktops, including datastore roles	Linked-clone OS disk storage: /RTP/ <b>host</b> /Desktop-Cluster1/Tier2-SAS Linked-clone persistent (user) data disk storage: /RTP/ <b>host</b> /Desktop-Cluster1/Tier2-SAS2 Linked-clone replica disk storage: /RTP/ <b>host</b> /Desktop-Cluster1/Tier2-SAS3 For full-clone desktops, each datastore serves the same role.
Linked clone desktops' parent VM, including path	/RTP/ <b>vm</b> /Master/Win7x32-LC
vCenter datacenter	RTP
vCenter Server name	vc01.vjason.local
View vCenter AD account	vjason\svc-view
View Composer Server AD account	vjason\svc-composer
View Composer AD domain	vjason.local
View Connection Server used	Viewmgr-01.vjason.local
View Finance Users AD Group	Finance_View_Users
View folder for the Finance pool	Finance
Virtual machine Snapshot 1	0222
Virtual machine Snapshot 2	0224
Virtual machine template for Full Clone desktops, with path	/RTP/ <b>vm</b> /Master/Win7x32-FC
vSphere cluster for desktops, with path	/RTP/ <b>host</b> /Desktop-Cluster1

Object name	Value
vSphere folder where the View Transfer Server is located	/RTP/ <b>vm</b> /Servers
vSphere resource pool for desktops, with path	/RTP/ <b>host</b> /Desktop-Cluster1/ <b>Resources</b>
vSphere Windows Customization Specification	View_Full_Clones

## View PowerCLI commandlets

The following is a list of all of the Horizon View PowerCLI commandlets. These commands can only be executed against a single View environment or pod at a time; if you have multiple distinct View environments, you must use separate PowerShell sessions for each. While each of these commandlets uses capital letters to identify individual words within the command, PowerCLI itself is not case-sensitive. You do not need to capitalize any part of the PowerCLI commands or the command options:

- `Add-AutomaticLinkedClonePool`: Creates an automatically provisioned linked-clone desktop pool
- `Add-AutomaticPool`: Creates an automatically provisioned full clone desktop pool
- `Add-ManualPool`: Creates a manually provisioned desktop pool of machines that can be managed by View
- `Add-ManualUnmanagedPool`: Creates a manually provisioned desktop pool of machines that cannot be managed by View
- `Add-PoolEntitlement`: Entitles a desktop pool using AD user or group names
- `Add-TerminalServerPool`: Creates a desktop pool that uses Windows Terminal Servers
- `Add-TransferServer`: Links a Transfer Server to the View environment
- `Add-ViewVC`: Links a vCenter Server to your View environment
- `Export-NetworkLabelSpecForFullClone`: Creates a network label specification file for a full-clone desktop pool
- `Export-NetworkLabelSpecForLinkedClone`: Creates a network label specification file for a linked-clone desktop pool
- `Get-ComposerDomain`: Retrieves a list of your View Composer servers and the domains they are attached to

- `Get-ConnectionBroker`: Retrieves a list of your View connection brokers, which include Connection and Security servers
- `Get-DesktopPhysicalMachine`: Retrieves a list of physical machines that can be used in unmanaged desktop pools
- `Get-DesktopVM`: Retrieves a list of virtual machines that are managed by View
- `Get-EventReport`: Retrieves a list of View event reports
- `Get-EventReportList`: Retrieves a list of View event reports and their descriptions
- `Get-GlobalSetting`: Retrieves the global configuration of the View environment
- `Get-License`: Retrieves the View license information
- `Get-LocalSession`: Retrieves a list of the current View Local Mode desktop sessions
- `Get-Monitor`: Retrieves a list of the View infrastructure health monitors and their current status
- `Get-Pool`: Retrieves a list of the View desktop pools
- `Get-PoolEntitlement`: Retrieves a list of users or groups assigned to a desktop pool
- `Get-ProfileDisk`: Retrieves a list of all of the persistent (user) disks
- `Get-RemoteSession`: Retrieves a list of active remote View sessions
- `Get-TerminalServer`: Retrieves a list of Windows Terminal Servers that are registered with View
- `Get-TransferPackage`: Retrieves a list of View Transfer Server packages
- `Get-TransferServer`: Retrieves a list of View Transfer Servers
- `Get-TransferServerState`: Retrieves the current status of the View Transfer Servers
- `Get-User`: Retrieves a list of Active Directory users and groups
- `Get-ViewVC`: Retrieves a list of vCenter Servers linked to the View
- `New-TransferPackage`: Creates a new Transfer Server package
- `Remove-Pool`: Removes the specified desktop pool
- `Remove-PoolEntitlement`: Removes desktop pool entitlement
- `Remove-TransferPackage`: Removes a View Transfer Server package
- `Remove-TransferServer`: Unlinks a View Transfer Server from View
- `Remove-UserOwnership`: Removes user ownership to a desktop

- `Remove-ViewVC`: Unlinks a vCenter Server from the View environment
- `Send-LinkedCloneRebalance`: Rebalances a linked-clone desktop pool
- `Send-LinkedCloneRecompose`: Recomposes a linked-clone desktop pool
- `Send-LinkedCloneRefresh`: Refreshes a linked clone-desktop pool
- `Send-LocalSessionRollback`: Rolls back a Local Mode desktop to the last checked in state
- `Send-SessionDisconnect`: Disconnects a View Client session
- `Send-SessionLogoff`: Logs off a View Client session
- `Send-VMReset`: Resets a View desktop
- `Set-ImageRepository`: Configures the View Transfer Server repository
- `Set-License`: Configures the View license key
- `Set-TransferServerMaintMode`: Enables or disables maintenance mode for a View Transfer Server
- `Update-AutomaticLinkedClonePool`: Updates the settings of an automatic assignment linked-clone pool
- `Update-AutomaticPool`: Updates the settings of an automatic assignment pool
- `Update-ConnectionBroker`: Updates the settings of a View connection broker
- `Update-GlobalSetting`: Updates View global settings
- `Update-ManualPool`: Updates the settings of a manual assignment desktop pool
- `Update-ManualUnmanagedPool`: Updates the settings of a manual unmanaged pool
- `Update-TerminalServerPool`: Updates the settings of a terminal server pool
- `Update-UserOwnership`: Updates the owner of a desktop
- `Update-ViewVC`: Updates the settings of a vCenter Server that is currently linked to the View environment

## Managing desktop pools with PowerCLI

View PowerCLI provides a number of different commands that View Administrators can use to provision and administer desktop pools. Nearly every option is available, which enables View administrators to build their own scripts for automating common View tasks.

## Desktop pool configuration options

The following options are supported when creating desktop pools using View PowerCLI. Options that are specific to linked-clone desktop pools or Virtual Desktops are indicated:

- `AllowMultipleSessions`: This specifies whether a single user can have multiple concurrent client sessions within the same desktop pool. This setting only applies when using floating assignment pools.
- `AllowProtocolOverride`: This specifies whether users can override the default display protocol, which is allowed by default.
- `AutoLogoffTime`: This specifies if desktops will log off after disconnect, and if so, after how long (in minutes).
- `Composer_ad_id`: This specifies the ID of the composer server to use when provisioning linked-clone desktops. Rather than use this setting, use the `Get-ComposerDomain` command as shown in the examples.
- `CustomizationSpecName`: This specifies the name of the vSphere customization specification to be used when customizing desktops using Windows Sysprep. When creating full-clone desktop pools, a customization specification is required or Windows customization will need to be performed manually after the desktops have been deployed.
- `DataDiskLetter`: This specifies the drive letter for the linked-clone persistent (user) data disk. The letter *D* will be used by default. Once the desktops have been deployed, the data disk letter cannot be changed.
- `DataDiskSize`: This specifies the size in MB of the linked-clone persistent (user) data disk. The disk is *2048 MB* by default. Once the desktops have been deployed, the data disk size cannot be changed using View.
- `DataStorePaths`: This specifies which datastores to use for provisioning full-clone desktops. The full path to the datastore must be provided and the entire value should be contained within quotes, for example, `"/RTP/host/Desktop-Cluster1/Tier2-SAS;/RTP/host/Desktop-Cluster1/Tier2-SAS2;/RTP/host/Desktop-Cluster1/Tier2-SAS3"`.



- **DatastoreSpecs:** This specifies which datastores to use for provisioning linked-clone desktops, their storage overcommit level, and their function. Storage overcommit level options are `None`, `Moderate`, `Aggressive`, and `Conservative` (the default). The datastore functions are `replica`, `data`, and `OS`. Separate datastores using semicolons, and multiple datastores can be specified for a single configuration type. The full path to the datastore must be provided and the entire value should be contained within quotes, for example, "[Aggressive, OS] /RTP/host/Desktop-Cluster1/Tier2-SAS; [Aggressive, data] /RTP/host/Desktop-Cluster1/Tier2-SAS2; [Aggressive, replica] /RTP/host/Desktop-Cluster1/Tier2-SAS3".
- **Description:** This is used to provide a more detailed description of the desktop pool. This setting is optional.
- **DefaultProtocol:** This specifies the default display protocol from either RDP or PCoIP (the default).
- **DeletePolicy:** This specifies if a desktop will be deleted or refreshed when the user logs off. By default, the desktops are not deleted. Options include `DeleteOnUse` or `RefreshOnUse`. `RefreshOnUse` applies to linked-clone desktops only.
- **DisplayName:** This displays the name of the desktop pool for the View Client.
- **Disabled:** This is used to enable or disable the desktop pool, preventing clients from logging in.
- **FlashQuality:** This specifies the maximum quality allowed for Adobe Flash content. Options include `HIGH`, `LOW`, `MEDIUM`, and `NO_CONTROL` (the default).
- **FlashThrottling:** This specifies how often Adobe Flash should refresh what it is displaying. Options include `AGGRESSIVE` (2500 ms), `CONSERVATIVE` (100 ms), `MODERATE` (500 ms), and `DISABLED` (the default).
- **FolderId:** This specifies which folder to place the desktop pool in within View, which is often used when delegating View administrative access at the View folder level, for example, `Finance`
- **HeadroomCount:** This specifies the minimum number of desktops to have powered on and available at all times.
- **IsProvisioningEnabled:** This specifies whether the pool will provision desktops when it is created. Options are `$true` or `$false`.
- **IsUserResetAllowed:** This specifies whether users can restart their Virtual Desktops. Users cannot reset their desktops by default. Options are `$true` or `$false`.

- `LogoffScript`: This specifies a script to run when someone logs off of a linked-clone desktop. Provide the full path to the script on the desktop.
- `MaximumCount`: This specifies the maximum number of desktops to create.
- `MaxVMsPerNetworkLabel`: This indicates the maximum number of desktops that can be created on each virtual machine network. It is used when creating a network label specification file.
- `MinimumCount`: This specifies that desktops should be provisioned on demand, in amounts equal to the number provided. To provision all desktops up front, use the same value specified for `MaximumCount`.
- `MinProvisionedDesktops`: This indicates the minimum number of linked-clone desktops that should be provisioned and available during View Composer maintenance operations. This value must be smaller than the `MinimumCount` value.
- `NamePrefix`: This specifies the name of the Virtual Desktops. Examples are:
  - `ViewDesktop{n}` or `ViewDesktop{n:fixed=3}`
    - The `fixed=3` specifies that the number must contain three digits.
- `NetworkLabelConfigFile`: This is used to configure the desktop virtual machine network and the number of desktops per virtual machine network, based on the specified network label specification file. The configuration file must have been previously exported.
- `OrganizationalUnit`: This indicates the AD DN of the OU in which to place the AD computer accounts. Do not include the DN of the domain itself. For example, `OU=ViewDesktops,OU=Corporate`
- `ParentSnapshotPath`: This indicates the path to the snapshot relative to the linked clone parent VM. You must insert a `/` before the snapshot name, for example, `/0222`.
- `ParentVmPath`: This indicates the path to the linked-clone parent virtual machine. You must include the full path, including the `vm` folder that is invisible within vCenter, for example, `/RTP/vm/Master/Win7x32-LC`.
- `Persistence`: This specifies whether the linked-clone desktops will be persistent or non-persistent. Options are `persistent` or `nonpersistent`.
  - When specifying `nonpersistent`, you cannot specify any of the options for persistent (user) data disks, as they cannot be used with non-persistent desktops. The pool will also be assigned floating user assignment.

- `Pool_id`: This indicates the View desktop pool ID. Spaces are not allowed when specifying the pool ID.
- `PostSyncScript`: This specifies a script to run when linked-clone desktops are provisioned, refreshed, or recomposed. Provide the full path to the script on the desktop.
- `PowerPolicy`: This specifies the power state of desktops when not in use. Options include `RemainOn`, `PowerOff`, `Suspend`, and `Always On` (the default).
- `ResourcePoolPath`: This indicates the path to the resource pool that will contain the virtual desktops. If resource pools are not being used, simply specify the vSphere cluster. You must include the full path, including the host and resources folders which are invisible within vCenter, for example, `/RTP/host/Desktop-Cluster1/Resources`.
  - If you are using a resource pool, specify it after resources in the path name, for example, `/RTP/host/Desktop-Cluster1/Resources/ResourcePool1`.
- `RefreshPolicyDays`: When linked-clone refresh after logoff is enabled, this specifies how often it will occur (in days).
- `RefreshPolicyType`: This specifies whether or not a linked-clone refresh will occur when a user logs off. Options are `Always`, `Conditional`, and `Never` (the default). When selecting `Conditional`, you must set either the `RefreshPolicyDays` or `RefreshPolicyUsage` parameters.
- `RefreshPolicyUsage`: When linked-clone refresh after logoff is enabled, this specifies how full the OS disk must be in percent before a refresh will be performed. Do not include the % sign when specifying a value.
- `SeSparseThreshold`: This specifies how much unused space in GB must exist on a linked-clone desktop before View will attempt to reclaim the space. The threshold is 1 GB by default.
- `SuspendProvisioningOnError`: This specifies whether View will suspend the provisioning of desktops within a pool if an error is encountered. Options are `$false` and `$true` (the default).
- `TempDiskSize`: This specifies the size in MB of the linked-clone temp (disposable) disk. The disk is 4096 MB by default.
- `TemplatePath`: This indicates the path to the template used to create full clone virtual desktops. You must include the full path, including the `vm` folder which is invisible within vCenter, for example, `/RTP/vm/Master/Win7x32-FC`.

- `UseSeSparseDiskFormat`: This specifies whether the **Space-Efficient** sparse virtual disks will be used, which enable View to reclaim stale or stranded data in the virtual machine disks and subsequently reduce storage space utilization.
- `UseTempDisk`: This specifies whether or not to use a linked-clone temp (disposable) disk. Options are `$false` and `$true` (the default).
- `UseUserDataDisk`: This specifies whether or not to use a linked-clone persistent (user) data disk. Options are `$false` and `$true` (the default).
- `Vc_id`: This specifies the ID of the vCenter Server to use when provisioning the desktops. Rather than use this setting, use the `Get-ViewVC` command as shown in the examples.
- `VmFolderPath`: This specifies the vCenter folder in which we have to create the virtual machines. If no folder is specified, View will create the folder for the desktop pool in the root of the vCenter folder hierarchy. The full path to the folder must be specified, including the invisible `vm` folder, for example, `/RTP/vm/Desktops`.

## Retrieving a list of the vCenter Servers linked to the View environment

The `Get-ViewVC` command supports the use of multiple options for retrieving vCenter Server information. These options include:

- `ComposerPort`: Port to use with View Composer Server
- `ComposerURL`: The View Composer URL in the `https://ComposerFQDN:Port` format
- `ComposerUsername`: The View Composer user name in `domain\username` format
- `Description`: Description for vCenter Server in View console
- `DisplayName`: Display name for vCenter Server in View Manager Admin console
- `Name`: View Composer Server name
- `Port`: Port to use with vCenter Server
- `ServerName`: View Composer Server name
- `Username`: The View Composer user name in `domain\username` format

---

The sample command retrieves View Composer Server information for the specified View Composer server:

```
Get-ViewVC -Name VC-01.vjason.local
```

Omit the options to retrieve a list of all vCenter Servers.

## Retrieving View Composer Server information

The `Get-ComposerDomain` command can be used to obtain View Composer information using the `Vc_id`, `Domain` or `Username` options. The sample command retrieves View Composer information based on which vCenter Server Composer is linked to:

```
Get-ComposerDomain -Vc_id (Get-ViewVC -Name VC-01.vjason.local).vc_id
```

The command `Get-ViewVC` is run within the command to obtain the `vc_id` value, which is easier than attempting to type the value in manually, as it is a series of random letters and numbers. This technique will be used in many of the examples for this chapter, as it makes working with certain values much easier. Omit the options to retrieve a list of all View Composer information.

## Network label specifications

A network label specification is used to configure desktop pools that need to place desktops on multiple virtual machine networks. This feature is currently available only when using View PowerCLI. Take a look at the following example.

Your vSphere cluster, where your View desktops will be deployed, has two virtual machine networks named *VLAN100* and *VLAN200*, and each virtual machine network can support no more than 250 desktops. You are tasked to create a desktop pool that has 500 desktops. To do this using the View Manager Admin console, you would need two Virtual Desktop master images, one connected to each virtual machine network. You would then need to create two desktop pools, one for each Virtual Desktop master image.

When you use View network label specification files, you need only one Virtual Desktop master image and one desktop pool. In this section, we will export our network label specification files, which we can then use when creating desktop pools.

The following screenshot shows the contents of a network label configuration file that will create at most 250 desktops in each of the two virtual machine networks:

```
#Network Label Configuration Spec
#WARNING! Setting enabled flag to false will
#turn off the automatic network label assignment
#For newly provisioned desktops.
enabled=true

#Parameter Definition for NIC
nic1=Network adapter 1

#Parameter Definition for Network
network01=VLAN100
network02=VLAN200

#Network Label Attribute Definition
#Expected format:
#<nic_param>.<network_param>.maxvm=<max vm for network label>
###nic1.network01.maxvm=250
###nic1.network02.maxvm=250
```

## Exporting network label specifications for linked-clone pools

The `Export-NetworkLabelSpecForLinkedClone` command requires several options in order to create the network label specification file including:

- `ClusterPath`: Indicates the Full path to the vSphere cluster
- `MaxVMsPerNetworkLabel`: Indicates the maximum number of desktops to place on each virtual machine network
- `ParentVmPath`: Indicates the full path to the parent VM used to export the network label data from
- `NetworkLabelConfigFile`: Indicates the location and filename of the exported network label specification file
- `ParentSnapshotPath`: Indicates the path to the parent VM snapshot, including
- `vc_id`: Indicates the vCenter Server ID

The sample command reads the network labels of the virtual machine networks for the specified linked-clone parent VM, located in the specified vSphere cluster and vCenter Server. The maximum number of VMs that will be created per network label is 250 and the network label specification file will be created on D:.

```
Export-NetworkLabelSpecForLinkedClone -ClusterPath /RTP/host/Desktop-Cluster1 -vc_id (Get-ViewVC -ServerName vc-01.vjason.local).vc_id -parentvmPath /RTP/vm/Master/Win7x32-LC -ParentSnapshotPath /0222 -MaxVMsPerNetworkLabel 250 -NetworkLabelConfigFile d:\LCConfigFile
```

The command `Get-ViewVC` is run within the command to obtain the `vc_id` value. You may also add the `FailIfNoNetworkFound` command option which will cause the command to fail if no suitable network labels are found on the vSphere cluster. The options are `$False` and `$True` (the default).



Since we are running a PowerCLI command on a remote system, the network label specification file will actually be created on D: of the View Connection Server.

## Exporting network label specifications for full-clone pools

The `Export-NetworkLabelSpecForFullClone` command requires the same input as the `Export-NetworkLabelSpecForLinkedClone` command. However, since we will be working with Virtual Desktop master images in the vSphere template format, we will be using the `TemplatePath` option and not the `ParentVmPath` and `ParentSnapshotPath` options. The `TemplatePath` is the full path to the Parent vSphere template used to provision the full-clone Virtual Desktops, including the invisible `vm` folder.

The sample command uses the same parameters as the `Export-NetworkLabelSpecForLinkedClone` command:

```
Export-NetworkLabelSpecForFullClone -ClusterPath /RTP/host/Desktop-Cluster1 -vc_id (Get-ViewVC -ServerName vc-01.vjason.local).vc_id -TemplatePath /RTP/vm/Master/Win7x32-FC -MaxVMsPerNetworkLabel 250 -NetworkLabelConfigFile d:\FCConfigFile
```

`Get-ViewVC` was run within the command to obtain the vCenter ID (`vc_id`).

## Creating an automatically provisioned linked-clone desktop pool

The commands used to create a (persistent) dedicated assignment and (non-persistent) floating assignment desktop pool are similar, so both are shown in this section.

## Creating a dedicated assignment (persistent) linked-clone pool

The `Add-AutomatedLinkedClonePool` command is used to create View linked-clone desktop pools. Some desktop pool options, such as View Storage Accelerator settings or some of the advanced options for SeSparse reclamation, cannot be configured using PowerCLI. These settings must be configured after the pool has been created using the View Manager Admin console.

The sample command will create a linked-clone desktop pool with the following characteristics:

- The pool will be created using the `VC-01.vjason.local` vCenter Server.
- The pool display name is `Finance Desktops`.
- The View pool ID is `FinanceLC1`.
- The desktop names will start with `FinanceLC`, and then be followed by a four-digit number.
- The desktops will be created in the vSphere cluster named `Desktop-Cluster1`. This cluster has no resource pools created within it, so the cluster itself will be used as the resource pool.
- The desktop pool folder will be placed in the `Desktops` folder within vCenter.
- The desktop pool will be placed in the `Finance` folder within the View Manager Admin console.
- The pool will contain 100 provisioned desktops, a minimum of 25 desktops will be available during View Composer maintenance operations, and 90 desktops will be powered on and available at all times.
- The View Composer Server is standalone in the test environment, so the dedicated user account for View Composer is specified in the `Get-ComposerDomain` command (`vjason\svc-composer`). If View Composer is installed directly on the vCenter Server, the vCenter Server user would be used instead (`vjason\svc-view`).
- Since no option is specified, the pool will be created as persistent desktops.
- The parent virtual machine, also known as the Virtual Desktop master image, is named `Win7x32-LC`, and the snapshot used to create the desktop pool is named `0222`.
- A network label configuration file will be used. This file is located on the virtual machine where the PowerCLI command is executed from, in the location specified.
- Since a vSphere Windows Customization Specification template is not named, the desktops will be customized using VMware QuickPrep.



- A different datastore is used for the OS, persistent (user) data disks, and replica disks, and the overcommit policy is set to `Aggressive` for each.
- The persistent (user) data disk uses letter `D` and is 1536 MB in size.
- The disposable data disk is 3072 MB in size.

When creating linked-clone desktop pools, you must specify the vCenter and View Composer domain in separate commands prior to beginning the command that actually creates the pool, for example:



```
Get-ViewVC -serverName vc-01.vjason.local | Get-ComposerDomain -domain vjason.local -username vjason\svc-composer |
```

The `|` character is used to feed the results of one PowerCLI command into the next command, an operation known as **piping**.

Not all of the values in the sample command are mandatory; the `FolderId`, `DataDiskLetter`, `DataDiskSize`, and `TempDiskSize` values can all be omitted and the View defaults will be used. The remaining values are all required in order to create a linked-clone pool using PowerCLI.

```
Get-ViewVC -serverName vc-01.vjason.local | Get-ComposerDomain -domain vjason.local -username vjason\svc-composer | Add-AutomaticLinkedClonePool -Pool_id FinanceLC1 -DisplayName "Finance Desktops" -NamePrefix "FinanceLC{n:fixed=4}" -VmFolderPath /RTP/vm/Desktops -ResourcePoolPath /RTP/host/Desktop-Cluster1/Resources -ParentVmPath /RTP/vm/Master/Win7x32-LC -ParentSnapshotPath /0222 -DatastoreSpecs "[Aggressive,OS]/RTP/host/Desktop-Cluster1/Tier2-SAS; [Aggressive,data]/RTP/host/Desktop-Cluster1/Tier2-SAS2; [Aggressive,replica]/RTP/host/Desktop-Cluster1/Tier2-SAS3" -maximumCount 100 -minprovisioneddesktops 25 -headroom 90 -minimumcount 100 -DataDiskLetter D -DataDiskSize 1536 -TempDiskSize 3072 -FolderId Finance -NetworkLabelConfigFile d:\LCConfigFile
```

## Creating a floating assignment (non-persistent) linked-clone pool

To make the linked-clone desktop pool floating non-persistent, the following changes would need to be made to the sample command from the previous section:

- Omit the options for `DataDiskLetter` and `DataDiskSize`.
- The OS and data disks must be same datastore, so adjust the datastore specifications to read `[Aggressive,data,OS]`.
- Add the `-Persistence NonPersistent` option.

Based on those requirements, the updated command is as follows. Look for [Aggressive,data,OS] and -Persistence NonPersistent; these are items that were added or changed. The items that were removed are not shown:

```
Get-ViewVC -serverName vc-01.vjason.local | Get-ComposerDomain -domain
vjason.local -username vjason\svc-composer | Add-AutomaticLinkedClonePool
-Pool_id FinanceLC1 -DisplayName "Finance Desktops" -NamePrefix
"FinanceLC{n:fixed=4}" -VmFolderPath /RTP/vm/Desktops -ResourcePoolPath /
RTP/host/Desktop-Cluster1/Resources -ParentVmPath /RTP/vm/Master/Win7x32-
LC -ParentSnapshotPath /0222 -DatastoreSpecs "[Aggressive,data,OS]/RTP/
host/Desktop-Cluster1/Tier2-SAS; [Aggressive,data,OS]/RTP/host/Desktop-
Cluster1/Tier2-SAS2; [Aggressive,replica]/RTP/host/Desktop-Cluster1/
Tier2-SAS3" -maximumCount 100 -minprovisioneddesktops 25 -headroom 90
-minimumcount 100 -TempDiskSize 3072 -FolderId Finance -Persistence
NonPersistent
```

## Creating an automatically provisioned full-clone desktop pool

The Add-AutomatedPool command is used to create View full-clone desktop pools. Some desktop pool options, such as View Storage Accelerator, cannot be configured using PowerCLI. These settings must be configured after the pool has been created using the View Manager Admin console.

The sample command will create a full-clone desktop pool using many of the same settings as the linked-clone pool, with the following changes:

- The pool display name is `Finance Desktops`
- The View pool ID is `FinanceFC1`
- The desktop names will start with `FinanceFC`, and then be followed by a four-digit number
- The parent vSphere template, also known as the Virtual Desktop master image, is named `Win7x32-FC`
- A vSphere Windows Customization Specification template named `view_Full_Clones` was specified, so Windows Sysprep will be used to customize the virtual machines
- Three datastores will be used to store the full clone desktops

Not all of the values in the sample command are mandatory; the `FolderID` and `CustomizationSpecName` can both be omitted and the View defaults will be used. The remaining values are all required in order to create a full-clone pool using PowerCLI:

```
Get-ViewVC -serverName vc-01.vjason.local | Get-ComposerDomain
-domain vjason.local -username vjason\svc-composer | Add-AutomaticPool
-Pool_id FinanceFC1 -DisplayName "Finance Desktops" -NamePrefix
"FinanceFC{n:fixed=4}" -VmFolderPath /RTP/vm/Desktops -ResourcePoolPath
/RTP/host/Desktop-Cluster1/Resources -TemplatePath /RTP/vm/Master/
Win7x32-FC -DatastorePath "/RTP/host/Desktop-Cluster1/Tier2-SAS;/RTP/
host/Desktop-Cluster1/Tier2-SAS2;/RTP/host/Desktop-Cluster1/Tier2-SAS3"
-maximumCount 100 -headroom 90 -minimumcount 100 -FolderId Finance -
CustomizationSpecName View_Full_Clones
```

## Creating a manually provisioned desktop pool

Manual desktop pools support most of the same configuration options as linked-clone or full-clone pools, as well as the following additional options:

- `Id`: vCenter machine ID for the virtual machine to be added to the pool
- `VC_name`: Host name of the vCenter Server that manages the pool VMs
- `Vm_id_list`: The ID for multiple virtual machines to add to the pool, separated by semicolons

The `Add-ManualPool` command requires at least the `Pool_id`, `VC_name` or `Vc_id`, and `Id` options to be specified to create a pool. For example:

```
Add-ManualPool -Pool_id Manual11 -Id (Get-DesktopVM -Name Desktop-01).id
-Vc_name VC-01.vjason.local
```

`Get-DesktopVM` was run within the command to obtain the value for the virtual machine ID (`id`).

## Creating a manual unmanaged desktop pool

Manual desktop pools support most of the same configuration options as linked-clone or full-clone pools, as well as the following additional options:

- `Pm_id`: Physical machine ID used to identify the physical desktop
- `Pm_id_list`: Physical machine ID used to identify multiple physical desktops, separated by semicolons

The `Add-ManualUnmanagedPool` command is similar to the `Add-ManualPool` command, but relies on **Windows System Identifiers (SIDs)** rather than vCenter machine IDs. For example:

```
Add-ManualUnmanagedPool -Pool_id Manual12 -Pm_id (Get-DesktopPhysicalMachine -Name Desktop-01).sid
```

`Get-DesktopPhysicalMachine` was run within the command to obtain the value for the Windows SID (`Pm_id`).

## Creating a Terminal Server desktop pool

The `Add-TerminalServerPool` command uses similar options as the `Add-ManualUnmanagedPool` command. For example:

```
Add-TerminalServerPool -Pool_id TermServ1 -Pm_id (Get-DesktopPhysicalMachine -Name TermServ-01).sid
```

`Get-DesktopPhysicalMachine` was run within the command to obtain the value for the Windows SID (`Pm_id`).

## Retrieving a list of the View desktop pools

The `Get-Pool` command can be used to obtain information on desktop pools based on the options `Description`, `DisplayName`, `Enabled` (`$true` or `$false`), `Pool_id`, `PoolType`, `Protocol`, and `VcServerName`. `VcServerName` is simply the name of the vCenter Server that hosts the desktop pool virtual machines. For example:

```
Get-Pool -Enabled $true -Protocol PCoIP
```

Omit the options to retrieve a list of all View desktop pools.

## Removing desktop pools

In this example, we will remove the desktop pool we created earlier using the `Remove-Pool` command. The only option required is the value for `Pool_id`.

```
Remove-Pool -Pool_id FinanceLC1
```

## Managing user entitlements

Four different View PowerCLI commands can be used to manage desktop or desktop pool entitlements. These commands are `Add-PoolEntitlement`, `Remove-PoolEntitlement`, `Update-UserOwnership`, and `Remove-UserOwnership`.

---

## Retrieving AD user or group information

The `Get-User` command is typically used to feed user or group names into other View PowerCLI commands. The following options are available:

- `IncludeUser`: This sets whether the results include AD user accounts. Options are `$False` and `$True` (the default).
- `IncludeGroup`: This sets whether the results include AD groups. Options are `$False` and `$True` (the default).
- `Name`: This indicates the name of the user or group to return. Partial matches are allowed based on the start of the name.
- `Domain`: This returns users or groups from a specific domain.

The following example returns only those AD groups that start with `view`:

```
Get-User -IncludeUser $false -Name View
```

Omit the options to retrieve a list of all users and groups.

## Entitling a desktop pool

The `Add-PoolEntitlement` and `Remove-PoolEntitlement` commands require you to specify the user or group AD system identifier (SID) in order to add or remove desktop pool entitlements. To do this, use the `Get-User` command within the `PoolEntitlement` command. Despite the name, the `Get-User` name is used to obtain both AD users and groups. For example:

```
Add-PoolEntitlement -Pool_id FinanceLC1 -sid (Get-User -Name "Finance_View_Users").sid
```

To entitle individual users, simply provide the first and last name of the user. For example:

```
Add-PoolEntitlement -Pool_id FinanceLC1 -sid (Get-User -Name "Jason Ventresco").sid
```

The `Remove-PoolEntitlement` command uses the same format as the `Add-PoolEntitlement` command; however, if you are removing the last entitlements from the desktop pool, you must add the option `-ForceRemove $true` for the command to succeed. This prevents you from accidentally removing all entitlements from a desktop pool. For example:

```
Add-PoolEntitlement -Pool_id FinanceLC1 -sid (Get-User -Name "Finance_View_Users").sid -ForceRemove $true
```

## Entitling or unentitling an individual desktop

Entitling an individual desktop is similar to entitling a desktop pool, except in this case we need both the user SID as well as the machine ID. For this command, we will nest two commands within the `Update-UserOwnership` and `Remove-UserOwnership` commands, `Get-DesktopVM` and `Get-User`. For example:

```
Update-UserOwnership -Machine_id (Get-DesktopVM -Name ViewLC0001).  
machine_id -Sid (Get-User -Name "Jason Ventresco").sid
```

The `Remove-UserOwnership` command requires only the desktop machine ID. For example:

```
Remove-UserOwnership -Machine_id (Get-DesktopVM -Name ViewLC0001).  
machine_id
```

## Reviewing desktop pool entitlement

The `Get-PoolEntitlement` command supports only one option, `Pool_id`. The sample command retrieves the entitlement settings for the desktop pool `FinanceLC1`:

```
Get-PoolEntitlement - Pool_id FinanceLC1
```

Omit the options to retrieve a list of user entitlements for all desktop pools.

## Updating the configuration of a desktop pool

Desktop pools are updated using the same options used when creating them. It is important to note that when performing a task such as adding an additional resource to the desktop pool, you must include any previous values in your calculation, such as the number of desktops or list of datastores. For example, if you have specified only a single new datastore when updating the linked-clone desktop pool configuration, the next time you perform maintenance on the desktop pool only the new datastore would be used.

## Updating a linked-clone pool

In this example, we will update the linked-clone desktop pool configuration using the `Update-AutomaticLinkedClonePool` command. The only option required is the value for `Pool_id`, and any options you wish to change. For example:

```
Update-AutomaticLinkedClonePool -Pool_id FinanceLC1 -  
AllowProtocolOverride $false
```

## Updating a full-clone pool

In the example provided, we will update the full-clone desktop pool configuration using the `Update-AutomaticPool` command. The only option required is the value for `Pool_id`, and any options you wish to change:

```
Update-AutomaticPool -Pool_id FinanceFC1 -DefaultProtocol RDP
```

## Updating a manually provisioned pool

The following example updates the manual pool configuration using the `Update-ManualPool` command. The only option required is the value for `Pool_id`, and any options you wish to change:

```
Update-ManualPool -Pool_id Manual1 -FlashQuality HIGH
```

## Updating a manually provisioned unmanaged pool

The manual unmanaged pool configuration can be updated using the `Update-ManualUnmanagedPool` command, as shown in the following example. The only option required is the value for `Pool_id`, and any options you wish to change:

```
Update-ManualUnmanagedPool -Pool_id Manual2 -FlashThrottling AGGRESSIVE
```

## Updating a Terminal Server pool configuration

The following example demonstrates how to update the manual pool configuration using the `Update-ManualPool` command. The only option required is the value for `Pool_id`, and any options you wish to change:

```
Update-TerminalServerPool -Pool_id TermServ1 -AutoLogoffTime 15
```

## View desktop maintenance

View PowerCLI includes multiple commands for reviewing virtual machine status and performing maintenance.

## Retrieving a list of virtual machines that are managed by View

The `Get-DesktopVM` supports multiple options to enable you to return desktops based on very specific parameters. The options include:

- `ComposerTask`: This retrieves desktops with the specified scheduled composer tasks. Options are `attachUdd`, `detachUdd`, `mkChkPoint`, `rebalance`, `refresh`, `replaceUdd`, and `resync`. **Udd** stands for **user data disk**.
- `GetNetworkLabel`: This retrieves the network label settings. Options are `$true` or `$false`.
- `IsInPool`: This retrieves desktops based on whether or not they are in a desktop pool. Options are `$true` or `$false`.
- `IsLinkedClone`: This retrieves desktops based on whether or not they are a linked clone. Options are `$true` or `$false`.
- `Name`: This displays the name of the desktop in vCenter.
- `Pool_id`: This indicates the desktop pool ID.
- `PoolType`: This lists only VMs that will work with the specified pool type. Options include `Manual` or `TransferServer`.
- `Vc_id`: This indicates the vCenter Server ID.

The following sample command retrieves a list of desktops that currently have a refresh operation scheduled:

```
Get-DesktopVM -ComposerTask refresh
```

Omit the options to retrieve a list of all virtual machines.

## Retrieving a list of physical machines

The `Get-DesktopPhysicalMachine` command supports multiple options to enable you to retrieve physical desktops based on very specific parameters. The options include:

- `Description`: This gives a description of the physical machine.
- `DisplayName`: This displays name of the physical machine.
- `Hostname`: This indicates the DNS name.
- `sid`: This indicates the Windows machine SID.



- **State:** This indicates the machine state. Options include `AgentUnreachable`, `Available`, `Connected`, `Disconnected`, `ConfigurationError`, or `Validating`.
- **OS:** This indicates the operating system. Options include `XP`, `Vista`, `Win7`, or `Win8`.

The following sample command retrieves a list of all desktops running the Windows 7 OS:

```
Get-DesktopPhysicalMachine -OS Win7
```

Omit the options to retrieve a list of all physical machines.

## Retrieving a list of the Windows Terminal Servers registered with View

The `Get-TerminalServer` command uses similar options as the `Get-DesktopPhysicalMachine` command. For example:

```
Get-TerminalServer -Hostname TermServ-01.vjason.local
```

Omit the option to retrieve a list of all Windows Terminal Servers.

## Retrieving information about persistent data disks

The `Get-ProfileDisk` command supports several options for retrieving information about the persistent disks registered with View:

- **Name:** This indicates the name of the persistent disk.
- **Username:** This indicates `FullDomain\username` of the owner of the persistent disk.
- **VmName:** This indicates the name of the VM that is using the persistent disk.
- **LastPool:** This indicates the desktop pool containing the persistent disk.
- **DataStore:** This indicates the datastore where the persistent disk is stored.
- **Status:** This indicates the status of the persistent disk. Options include `InUse`, `Archiving`, and `Detached`.

The sample command will retrieve information about the persistent disk that belongs to the specified user:

```
Get-ProfileDisk -Username vjason.local\jventresco
```

## Resetting a View desktop

The `Send-VMReset` command can be used to reset a View desktop, such as when it is in an unresponsive state. The command requires the machine ID in order to identify the desktop. For example:

```
Send-VMReset -Machine_id (Get-DesktopVM -Name ViewLC0001).machine_id
```

## Refreshing a linked-clone desktop or Pool

The `Send-LinkedCloneRefresh` command is used to refresh a linked-clone desktop or pool. When refreshing an entire pool, the command requires us to specify each desktop within the pool, so we will be piping the output of the `Get-Pool` and `Get-DesktopVM` commands into the `Send-LinkedCloneRefresh`. You must also specify the time to begin the refresh using the `-schedule` option in the format 'YYYY-MM-DD HH:MM' using a 24-hour format for the hour.

Other options for the command include `StopOnError`, which is enabled by default and halts the refresh if errors occur, and `ForceLogoff`, which is disabled by default and will force users to log off. Both of these options accept either `$true` or `$false` as options. For example:

```
Get-Pool -Pool_id FinanceLC1 | Get-DesktopVM | Send-LinkedCloneRefresh  
-schedule '2013-02-25 18:00' -StopOnError $false -ForceLogoff $false
```

This command selects all the desktops in the `FinanceLC1` pool, and schedules them to refresh at the indicated time. In addition, the operation will continue even if an error occurs, but will not force users to log off.

To refresh just a single desktop, you can use a simpler version of the command that requires only the machine ID and the schedule. For example:

```
Send-LinkedCloneRefresh -Machine_id (Get-DesktopVM -Name ViewLC0001).  
machine_id -schedule '2013-02-25 18:00'
```

This command will refresh only the desktop named `ViewLC0001`.

## Recomposing a linked-clone desktop pool

The `Send-LinkedCloneRecompose` command requires you to specify multiple options including `Schedule`, `ParentVMPATH`, and `ParentSnapshotPath`. The command also supports the `StopOnError` and `ForceLogoff` options.

We will be recomposing to a new snapshot named 0225 of the same parent VM. Since this VM now has two snapshots, the `ParentSnapshotPath` will now be in the format `/0222/0225`, where 0222 is the original snapshot used to create the pool. The remainder of the command follows a format similar to the `Send-LinkedCloneRefresh` command. For example:

```
Get-Pool -Pool_id FinanceLC1 | Get-DesktopVM | Send-LinkedCloneRecompose
-ParentVMPath /RTP/vm/Master/Win7x32-LC -ParentSnapshotPath /0222/0225
-schedule '2013-02-25 18:00'
```

The command will recompose all desktops in the pool to the snapshot named 0225 at the time indicated. You may also select a different parent VM when performing a recompose, but remember that the VM must be running the same OS as the existing desktops.

You can also recompose a single desktop using the `-machine_id` option and the `Get-DesktopVM` command. For example:

```
Send-LinkedCloneRecompose (Get-DesktopVM -Name ViewLC0001).machine_id
-ParentVMPath /RTP/vm/Master/Win7x32-LC -ParentSnapshotPath /0222/0225
-schedule '2013-02-25 18:00'
```

## Rebalancing a linked-clone desktop pool

The `Send-LinkedCloneRebalance` command uses the same format as the other linked-clone maintenance. All that is required is the desktop pool ID and the schedule. The command also supports the `StopOnError` and `ForceLogoff` options. For example:

```
Get-Pool -Pool_id FinanceLC1 | Get-DesktopVM | Send-LinkedCloneRebalance
-schedule '2013-02-25 18:00'
```

This command selects all the desktops in the `FinanceLC1` pool and schedules them to rebalance at the indicated time.

To rebalance just a single desktop, you can use a simpler version of the command that requires only the machine ID and the schedule. For example:

```
Send-LinkedCloneRebalance -Machine_id (Get-DesktopVM -Name ViewLC0001).
machine_id -schedule '2013-02-25 18:00'
```

This command will rebalance only the desktop named `ViewLC0001`.

## Rolling back a View local mode desktop

The `Send-LocalSessionRollback` command requires only the machine ID.

For example:

```
Send-LocalSessionRollback -Machine_id (Get-DesktopVM -Name ViewLC0001).  
machine_id
```

## Configuring the View environment using PowerCLI

There are a number of PowerCLI commands that can be used to configure and manage key components of the View infrastructure.

## Linking a vCenter Server to View

The `Add-ViewVC` command requires several options be specified in order to link a vCenter Server to the View environment. They include:

- `CreateRampFactor`: This indicates the maximum concurrent vCenter desktop provisioning operations.
- `Password`.
- `ServerName` or `Name`: This indicates the FQDN of the vCenter Server. Either option can be specified.
- `Username` or `User`: User with appropriate permissions within vCenter, in the format `domain\username`. Either option can be specified.

Additional options can be specified. These include:

- `ComposerPort`: This indicates the port to use with View Composer Server.
- `DeleteRampFactor`: This indicates the maximum concurrent desktop power operations.
- `Description`: This gives you a description for vCenter Server in View console.
- `DisplayName`: This displays the name for vCenter Server in View Manager Admin console.
- `Port`: This indicates the port to use with vCenter Server.
- `UseComposer`: Use View Composer Server installed on vCenter Server. Options are `$true` or `$false`.

- `UseComposerSSL`: Use SSL when connecting to the View Composer Server.
- `UseSpaceReclamation`: This enables SeSparse space reclamation on vSphere hosts managed by the vCenter Server.
- `UseSsl`: Use SSL when connecting to the vCenter Server.
- Options such as port numbers and whether or not to use SSL (enabled by default) will use their default values if not specified and should not be changed under most circumstances. A number of vCenter options cannot be configured when linking a vCenter Server using PowerCLI. These include View Storage Accelerator, standalone View Composer Servers, dedicated users for View Composer, View Composer domains, and others. These options can only be configured after the parent feature has been enabled using the View Manager Admin console.



If your vCenter Server or View Composer Server SSL certificate is not trusted by the View Connection Servers, the `Add-ViewVC` operation will fail. This is different from adding a vCenter Server using the View Manager Admin console, which allows you to accept an untrusted certificate. To use this command, you must replace the default vCenter Server SSL certificate with one signed by a trusted certificate authority.

The following example links the `vc-01.vjason.local` vCenter Server to View:

```
Add-ViewVC -ServerName vc-01.vjason.local -Username vjason\svc-view
-Password Password123 -CreateRampFactor 8 -UseComposer $true
```

## Updating the settings of a vCenter Server that is linked to View

The `Update-ViewVC` command can be used to update the settings of a vCenter Server that is currently linked to View. This command supports the same options as the `Add-ViewVC` command. Specify the vCenter Server to update using the `ServerName` or `Name` option and then update the options as required. For example:

```
Update-ViewVC -ServerName vc-01.vjason.local -DeleteRampFactor 10 -
Description "VC-01 vCenter Server"
```

## Unlinking a vCenter Server from View

The `Remove-ViewVC` requires only the vCenter Server name in order to unlink it from View. The vCenter Server cannot be removed if desktops are currently deployed.

For example:

```
Remove-ViewVC -ServerName vc-01.vjason.local
```

## Adding a View Transfer Server

The `Add-TransferServer` command requires two options to be specified in order to link a vCenter Server to the View environment. They include:

- `Vc_id`
- `Tsvm_path`: Full path to the Transfer Server virtual machine in vCenter

The following command will link the `VIEWTRAN-01.vjason.local` Transfer Server, located in the vCenter folder `Servers`, running on the `VC-01.vjason.local` vCenter Server to our View environment:

```
Add-TransferServer -Vc_id (Get-ViewVC -Name VC-01.vjason.local).vc_id -  
Tsvm_path /RTP/vm/Servers/VIEWTRAN-01
```

## Retrieving a list of Transfer Servers

The `Get-TransferServer` command supports only one option, `Path`. This is the full path to the Transfer Server in vCenter. For example:

```
Get-TransferServer -Path /RTP/vm/Servers/VIEWTRAN-01
```

Omit the option to retrieve a list of all Transfer Servers.

## Removing a Transfer Server

The `Remove-TransferServer` command supports only one option, `Ts_id`, which is the View ID for the Transfer server. Rather than type out the ID, we will use `Get-TransferServer` within the command to obtain the ID. For example:

```
Remove-TransferServer -Ts_id (Get-TransferServer -path /RTP/vm/Servers/  
VIEWTRAN-01).ts_id
```

---

## Setting the Transfer Server Maintenance mode

The `Set-TransferServerMaintMode` command requires two options, `Ts_id` and `Maintenance`. The `Maintenance` option accepts `$true` (enable maintenance mode) or `$false` (disable maintenance mode). For example:

```
Set-TransferServerMaintMode -Ts_id (Get-TransferServer -path /RTP/vm/Servers/VIEWTRAN-01).ts_id -Maintenance $true
```

## Retrieving the Transfer Server state

The `Get-TransferServerState` command supports only one option, `Ts_id`. For example:

```
Get-TransferServerState -Ts_id (Get-TransferServer -Path /RTP/vm/Servers/VIEWTRAN-01).ts_id
```

## Configuring a Transfer Server image repository

The `Set-ImageRepository` command requires multiple options in order to configure the View Transfer Server repository:

- `Local_repo`: This specifies whether the repository is located on the Transfer Server (`$true`) or is remote (`$false`).
- `Path`: This specifies the path to the repository. If local, supply a path to a folder on the Transfer Server; if remote, supply a UNC path to a remote share.
- `User_domain`: This indicates the user domain name when using a network (remote) repository.
- `User_name`: This indicates the username when using a network (remote) repository.
- `User_password`: This indicates the user password when using a network (remote) repository.

The following sample command configures a network repository for the Transfer Server:

```
Set-ImageRepository -Local_repo $false -Path \\FS-01.vjason.local\ViewTrans -User_domain vjason.local -User_name svc-transfer -User_password Password123
```

## Retrieving View license information

The `Get-License` command has no options; simply execute the command by itself to retrieve View license status.

## Configuring the View license

The `Set-License` command requires only one option, `Key`. Do not remove the dashes from the license key. For example:

```
Set-License -Key 11111-AAAAA-22222-BBBBB-33333
```

## Updating View Connection Broker settings

The `Update-ConnectionBroker` command supports a number of options for configuring View Connection Brokers, which include both Connection Servers and Security Servers:

- `Broker_id`: This indicates the name of the View connection broker.
- `DirectConnect`: This enables direct connections to View desktops.
- `DirectPCoIP`: This enables direct PCoIP connections to View desktops.
- `ClearNodeSecret`: This clears the RSA SecurID node secret.
- `ExternalURL`: This indicates the external URL for the Connection Server home page.
- `ExternalPCoIPUrl`: This indicates the external URL for PCoIP access using the secure gateway.
- `LdapBackupFolder`: This indicates the folder used for View LDAP backups.
- `LdapBackupFrequency`: This indicates the frequency of LDAP backups.
- `LdapBackupMaxNumber`: This indicates the maximum number of LDAP backups to retain.
- `LogoffWhenRemoveSmartCard`: This logs off View Client sessions when the user Smart Card is removed.
- `DirectConnectForLocal`: This allows direct connections to Local Mode desktops.
- `UseCompressionForLocal`: Uses compression for Local Mode desktop operations.
- `UseDedupForLocal`: Uses deduplication for Local Mode desktop operations.
- `UseSSLForLocal`: Uses SSL for Local Mode desktop operations. Enabled by default.



- `UseSSLForLocalProvisioning`: Uses SSL when provisioning Local Mode desktops. Enabled by default.
- `NameMapping`: This enforces RSA SecurID and Windows name matching.
- `SecureIDEnabled`: This enables RSA SecurID authentication.
- `SmartCardSetting`: This enables Smart Card authentication. Options are `Required`, `Off`, or `Optional` (the default).
- `Tags`: This sets Connection Server tags, which are used to restrict desktop pools to specific Connection Servers.
- `PCoIPBandwidthLimit`: This configures the per-session PCoIP bandwidth limit.

The following sample command updates the external PCoIP URL of the View Security Server named `VIEWSEC-01`:

```
Update-ConnectionBroker -Broker_id VIEWSEC-01 -ExternalPCoIPUrl  
192.168.0.1:4172
```

## Retrieving View Connection Broker information

The `Get-ConnectionBroker` command supports only one option, `Broker_id`, which is the computer name of the Connection or Security Server you wish to retrieve. Omit the options to retrieve all View connection brokers. The sample command retrieves information about the `VIEWMGR-01` Connection Server:

```
Get-ConnectionBroker -Broker_id VIEWMGR01
```

## Updating the View global settings

The `Update-GlobalSetting` command can be used to update a number of different View global settings. The following settings can be set using this command:

- `DisableLocalSSO`: This disables SSO for logging into Local Mode desktops. Options are `$true` or `$false`.
- `DisplayLogoffWarning`: This displays a warning to the View Client prior to a forced logoff.
- `DisplayPreLogin`: This displays a login message prior to the View Client logging into the Connection Server.
- `ForceLogoffAfter`: This sets how long to wait after the warning message appears to force logoff the View Client.

- `ForceLogoffMessage`: This indicates the text for the force logoff message.
- `MessageSecurityMode`: This sets the security level for communication between View components. Options include `Disabled`, `Mixed`, and `Enabled` (the default).
- `PreLoginMessage`: This indicates the text for the pre-login message.
- `ReauthenticateOnInterrupt`: This forces the View Client to reauthenticate after connection interruption. Options are `$true` or `$false`.
- `SessionTimeout`: This indicates the time-out value for inactive View Client sessions.
- `UseSslClient`: This forces SSL View Client connections. Options are `$true` or `$false`.
- `WidgetPolling`: This enables automatic status updates in the View Administrator. Options are `$true` or `$false`.

The following sample command enables and configures `ForceLogoffMessage` and `PreLoginMessage`:

```
Update-GlobalSetting -DisplayPreLogin $true -PreLoginMessage  
"Unauthorized users prohibited" -DisplayLogoffWarning $true -  
ForcedLogoffMessage "You will be logged off"
```

## Retrieving global View configuration data

The `Get-GlobalSetting` command has no options; simply execute the command by itself to retrieve information about the View global settings.

## Managing View Transfer Server packages

There are three different PowerCLI commands that can be used to manage View Transfer Server packages: `New-TransferPackage`, `Get-TransferPackage`, and `Remove-TransferPackage`. Transfer Server packages are View Composer base images that approved View Local Mode clients can download to their local machine.

## Configuring a New Transfer Server package

The `New-TransferPackage` command requires multiple options in order to configure a View Transfer Server package:

- `BaseVm_path`: Indicate the full path to the VM in vCenter used to create the package
- `Pool_id`

- `Snapshot_id`: The ID of the snapshot of the VM used to create the package
- `Vc_id`

The sample command creates a package using an existing desktop pool and base VM:

```
New-TransferPackage -BaseVm_path /RTP/vm/Master/Win7x32-LC -Snapshot_id /0222 -Pool_id FinanceLC1 -Vc_id (Get-ViewVC -Name VC-01.vjason.local).vc_id
```

## Retrieving information about a Transfer Server package

The `Get-TransferPackage` command supports only one option, `Package_id`. For example:

```
Get-TransferPackage -Package_id 537d66e1-35d8-4c1d-bde7-55a56e1afaa5
```

Omit the option to retrieve a list of all Transfer Server packages.

## Removing a Transfer Server package

The `Remove-TransferPackage` command supports only one option, `Package_id`. For example:

```
Remove-TransferPackage -Package_id 537d66e1-35d8-4c1d-bde7-55a56e1afaa5
```

## Monitoring the View environment using PowerCLI

View PowerCLI includes multiple commands that can be used to manage and monitor user sessions and View resources.

## Monitoring and managing View Client sessions

View PowerCLI includes four commands for managing View Client sessions: `Get-RemoteSession`, `Get-LocalSession`, `Send-SessionDisconnect`, and `Send-SessionLogoff`.

## Monitoring remote View sessions

The `Get-RemoteSession` command supports multiple options for listing client connections. Only one option is required to retrieve session information, `Pool_id`. The options include:

- `Username`: Indicates the username in the `FullDomainName\username` format, for example, `vjason.local\jventresco`
- `Pool_id`: Indicates the desktop pool ID, for example, `FinanceLC1`
- `Session_id`: Indicates the View session ID
- `Duration`: Indicates the duration in the format "*dd day(s) hh hour(s) mm minute(s) ss second(s)*", for example, "`2 days 1 hour 15 minutes 1 second`"
- `DnsName`: Indicates the DNS name of the Virtual Desktop
- `State`: Indicates the state of the desktop (`Connected` or `Disconnected`)
- `Protocol`: Indicates the protocol being used in session (`PCoIP` or `RDP`)
- `StartTime`: Indicates the time the session was started including the day, time, time zone, and year, for example, "`Mon Feb 25 16:00:15 EST 2013`"
- The following sample command retrieves all remote View sessions for the desktop pool `FinanceLC1`:

```
Get-RemoteSession -Pool_id FinanceLC1
```

## Monitoring local mode desktop sessions

The `Get-LocalSession` command supports the `Pool_id` and `Duration` options described in the `Get-RemoteSession` command. The command also supports the following additional options:

- `PoolDisplayName`: Displays the name of the desktop pool
- `Hostname`: Indicates the hostname of the desktop
- `LocalHostname`: Indicates the hostname of the View Client running the desktop
- `LocalUserDisplayName`: Indicates the username in the `FullDomainName\username` format
- `LocalState`: Indicates the state of the desktop (`Connected` or `Disconnected`)
- `Machine_id`: Indicates the machine ID obtained using the `Get-DesktopVM` command

- `Id`: Indicates the VM ID of the desktop in vCenter
- `VmDisplayName`: Displays the name of the desktop in vCenter

The following sample command retrieves session information for the local mode desktop belonging to the specified user:

```
Get-LocalSession -LocalUserDisplayName vjason.local\jventresco
```

## Disconnecting View Client sessions

The `Send-SessionDisconnect` command disconnects users based on the View session ID. The session ID is a really long value that is difficult to work with, so we will use the `Get-RemoteSession` command within the `Send-SessionDisconnect` command instead in order to disconnect the target user. For example:

```
Send-SessionDisconnect -Session_id (Get-RemoteSession -Username vjason.local\jventresco).session_id
```

## Logging off View Client sessions

The `Send-SessionLogoff` command uses the same format as the `Send-SessionDisconnect` command. For example:

```
Send-SessionLogoff -Session_id (Get-RemoteSession -Username vjason.local\jventresco).session_id
```

## Monitoring the View core infrastructure

View PowerCLI supports multiple commands for retrieving information about View events and the status of the View infrastructure itself.

## Retrieving View event reports and their descriptions

The `Get-EventReportList` command has no options; simply execute the command by itself to retrieve a list of View event report names and their description.

## Retrieving View event reports

The `Get-EventReport` command supports three options:

- `ViewName`: This indicates the name of the event report to output. Options include `config_changes`, `user_auth_failures`, `user_count_events`, and `user_events`.
- `StartDate`: This indicates the start date for the report.
- `End-Date`: This indicates the end data for the report.

The sample command retrieves all event data about user events:

```
Get-EventReport -ViewName user_events
```

## Retrieving View infrastructure health monitors and their statuses

The `Get-Monitor` command supports two different options:

- `Monitor_id`: This indicates the ID of the monitor. You can provide the specific monitor ID itself as obtained from the `Get-Monitor` command, or you can specify a View server name and all monitors for that server will be returned.
- `Monitor`: This indicates the name of the monitor. Possible values include:
  - `CBMonitor`: Indicates the Connection Server monitor.
  - `DBMonitor`: Indicates the View event database monitor.
  - `DomainMonitor`: Indicates the domain connection monitor.
  - `SGMonitor`: Indicates the Security Server monitor.
  - `TSMonitor`: Indicates the Transfer Server monitor.
  - `VCMonitor`: Indicates the vCenter Server monitor.

The sample command retrieves all monitoring data for the specified server:

```
Get-Monitor -Monitor_id VIEWMGR-01
```

Omit the options to retrieve a list of all View monitoring data.

## Summary

In this chapter, we learned about how to use View PowerCLI to manage VMware View. We discussed what is required to use PowerCLI remotely, how to enable WinRM on a View Connection server, learned what each PowerCLI commandlet is used for, and looked at examples of each of the PowerCLI commands.

In *Chapter 15, VMware Horizon View Feature Pack 1* (available for free download from the Packt Publishing website), we will discuss the new features introduced with Feature Pack 1, which include access to View desktops over HTML5-compliant browsers and the touch screen-optimized View Client Unity Touch interface.

