Double-Take®

Version 7.0
PowerShell Scripting Guide

VISION® SOLUTIONS

Notices

Double-Take PowerShell Scripting Guide Version 7.0, Friday, November 22, 2013

Check the Vision Solutions support web site at http://www.VisionSolutions.com/SupportCentral for the most up-to-date version of this documentation.

- **Product Updates**—Check your service agreement to determine which updates and new releases you may be eligible for. Product updates can be obtained from the support web site at http://www.VisionSolutions.com/SupportCentral.
- **Sales**—If you need maintenance renewal, an upgrade activation code, or other sales assistance, contact your reseller/distributor or a Vision Solutions sales representative. Contact information is available on the Vision Solutions Worldwide Locations and Contacts web page at http://www.VisionSolutions.com/Company/Vision-HA-Locations.aspx.
- **Technical Support**—If you need technical assistance, you can contact CustomerCare. All basic configurations outlined in the online documentation will be supported through CustomerCare. Your technical support center is dependent on the reseller or distributor you purchased your product from and is identified on your service agreement. If you do not have access to this agreement, contact CustomerCare and they will direct you to the correct service provider. To contact CustomerCare, you will need your serial number and activation code. Contact information is available on the Vision Solutions CustomerCare web page at http://www.VisionSolutions.com/Support/Support-Overview.aspx .
- **Professional Services**—Assistance and support for advanced configurations may be referred to a Pre-Sales Systems Engineer or to Professional Services. For more information, see the Windows and Linux tab on the Vision Solutions Consulting Services web page at http://www.VisionSolutions.com/Services/Consulting-Services.aspx .
- **Training**—Classroom and computer-based training are available. For more information, see the Double-Take Product Training web page at http://www.VisionSolutions.com/Services/DT-Education.aspx.
- **Documentation**—Please forward any comments or suggestions about this documentation to documentation-Double-Take@VisionSolutions.com.
- **Linux man pages**—Man pages are installed and available on Double-Take Linux servers. These documents are bound by the same Vision Solutions license agreement as the software installation. Check the support web site at http://www.VisionSolutions.com/SupportCentral for the most up-to-date version of these files.

Contents

Chapter 1 Overview

Double-Take includes Windows PowerShell cmdlets that you can use to control most Double-Take features. This guide includes all of the Double-Take cmdlets available and several sample scripts.However, this guide does not explain how to use Windows PowerShell. You should reference your Windows PowerShell documentation and the many web sites devoted to PowerShell to learn how to use and script with Windows PowerShell.

The Double-Take Availability agentless vSphere job does not support PowerShell. You cannot create, control, or manage agentless vSphere jobs with any Double-Take PowerShell cmdlets. Additionally, Double-Take Reporting Service does not support PowerShell. You cannot configure or manage your Double-Take Reporting Service server with any Double-Take PowerShell cmdlets.

Review the following terms and definitions to help you understand the Double-Take PowerShell cmdlets. See the Double-Take Availability *User's Guide* and the Double-Take Move *User's Guide* for complete details on how these products work.

- **Source**—The source is the server that has the data you want to protect. Typically this is a machine on the production network that serves data to clients.
- **Target**—The target is the server that maintains the replicated copy of the data that is being protected on the source. Typically this is a backup server that may be local or in a remote data center.
- **Workload**—A workload is a logical definition of the data that is being protected on the source. A workload can be a simple set of paths, for example, C:\Public, or it may be a more complex logical item that maps to multiple paths, for example, protecting a virtual machine means you are protecting multiple, specific virtual machine files.
- **Workload Manager**—The workload manager is a web service that creates and configures the Double-Take workload.
- **Job**—A job is a logical unit that includes the source, target, and the workload. The job is what you create and monitor in order to protect your data.
- **Job Manager**—The job manager is a web service that creates, monitors, and controls the Double-Take job.
- **Connection**—The engine connection, sometimes just called the connection, is the underlying stream that sends the actual replicated data between the source and target servers. Jobs are higher-level objects that use the lower-level connection to protect data.
- **Architecture**—Each Double-Take installation has two key services, the Management Service and Engine.
  - **Management Service**—This service is displayed as Double-Take Management Service in the services list. The service hosts the Job Manager and provides monitoring and control for all job types. By default this service listens on port 6325. This service offers a SOAP-based XML web services interface, implemented using WCF.
  - **Engine**—This service is displayed as Double-Take in the service list. The service transmits the replicated data between the source and target servers. By default this service listens on port 6320. You do not interact directly with this service.
- **Roles**—Any Double-Take installation can be a source, target, or both. The existence of a job between two servers and which direction data is being transmitted determines the server's role.
- **Security**—Double-Take enforces security by using local groups on each server where Double-Take is installed. There are two levels of security. Double-Take Admin allows full control of Double-Take on a

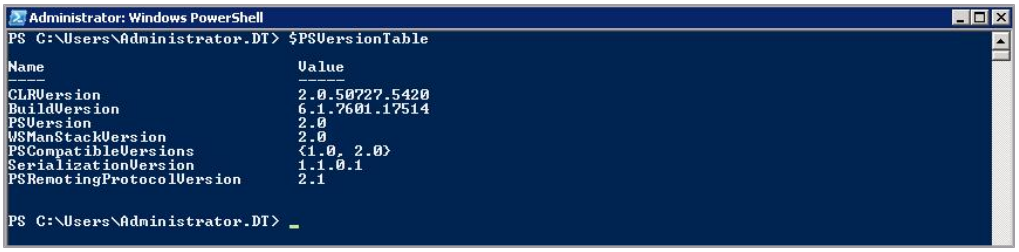
server, and Double-Take Monitor allows read-only access to job information. When you connect to the job manager on a server, you will need to provide the credentials of a user who is a member of one of the local groups on that server.

- **Job creation**—To create a job, you will first communicate with the source to create a workload. You will then use that workload object and communicate with the target to create the job.
- **Monitoring and controlling jobs**—To monitor and control jobs, you will communicate with the job manager on the target of the job.

Double-Take PowerShell requirements

Double-Take requires Windows PowerShell version 2 or later. This version is installed by default on newer Windows operating systems, like Windows 2008 R2 and Windows 7, or newer updates like Windows 2003 Service Pack 2. However, it has been a part of the Windows update functionality since June 2010, so you may have version 2 installed even if you are running an older Windows operating system.

If you are uncertain which version you have installed, check the PSVersion property of the $PSVersionTable automatic variable. This variable does not exist in PowerShell version 1, so if the variable returns nothing, you have version 1 installed. If you have version 2 installed, you will see a table of version information, showing your major and minor version numbers.

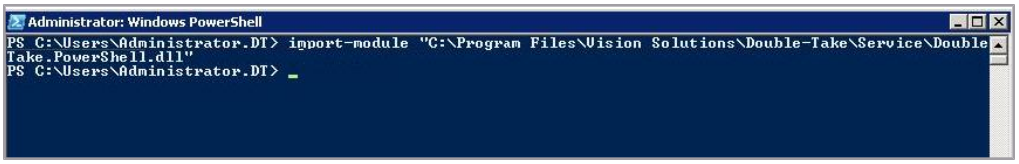


Installing the Double-Take PowerShell module

There are no additional steps required to install the Double-Take PowerShell module. It is automatically installed with all Double-Take installations.

Importing the Double-Take PowerShell module

You will need to import the module before you can begin using it. Use the Windows PowerShell import-module cmdlet to import the DoubleTake.PowerShell.dll module. If you completed a server or client/server installation, the module will be located in the \Service subdirectory where you installed Double-Take. If you completed a client only installation, the module will be located in the \Console subdirectory where you installed Double-Take. By default, the installation location is \Program Files\Vision Solutions\Double-Take. For example, using the default server installation lcoation, the cmdlet would be import-module "C:\Program Files\Vision Solutions\Double-Take\Service\DoubleTake.PowerShell.dll" or using the default client only installation location, the cmdlet would be import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll".



If nothing is returned, then the import cmdlet was successful.

The import-module cmdlet only imports a module into the current session. If you need to make the Double-Take PowerShell module available to all sessions, you will need to add an import-module cmdlet to your Windows PowerShell profile. See your PowerShell documentation for more information about profiles.

Chapter 2 Cmdlets

The Double-Take PowerShell cmdlets can be divided into five functional areas.

- **Server object**—Server objects cmdlets are used to create and manipulate Double-Take server objects that store name and credential information for contacting the Double-Take Management Service on a specified machine.
- **Server management**—Server management cmdlets are used to configure server settings and to invoke Double-Take actions on a server.
- **Workload**—Workload cmdlets are used to create and manipulate workloads. This is the data you are protecting on your source. You can have physical rules (volumes, folders, and files) or logical rules (SQL protection contains multiple rules to protect a SQL database and all of its associated files).
- **Job**—Job cmdlets are used to create, configure, and manage Double-Take jobs.
- **Engine control**—Engine control cmdlets are used to control the connection-level processing of a job.

Cmdlet	Server Object	Server Management	Workload	Job	Engine Control
Add-DtPhysicalRule on page 18			X		
Add-DtUvraPhysicalRule on page 20			X		
Checkpoint-DtConnection on page 22					X
Close-DtWorkload on page 24			X		
Confirm-DtJobOptions on page 25				X	
Disconnect-DtServer on page 28	X				
Edit-DtJob on page 29				X	
Get-DtAccessLevel on page 31		X			
Get-DtActivationStatus on page 32		X			
Get-DtBandwidthLimit on page 33					X
Get-DtConnectionIds on page 35					X
Get-DtDiagnostics on page 36		X			
Get-DtEmailNotificationOptions on page 37		X			
Get-DtEventLogEntry on page 38		X			
Get-DtJob on page 39				X	
Get-DtJobActionStatus on page 41				X	

Cmdlet	Server Object	Server Management	Workload	Job	Engine Control
Get-DtLogicalItem on page 43			X		
Get-DtLogMessage on page 44		X			
Get-DtOnlineActivationRequest on page 46	X				
Get-DtOption on page 47		X			
Get-DtPathBlocking on page 48		X			
Get-DtPhysicalItem on page 49		X			
Get-DtProductInfo on page 50		X			
Get-DtProxiedServerInfo on page 51				X	
Get-DtQualificationResults on page 52				X	
Get-DtRecommendedFailbackOptions on page 54				X	
Get-DtRecommendedFailoverOptions on page 56				X	
Get-DtRecommendedJobOptions on page 58				X	
Get-DtRecommendedPathTransform on page 61			X		
Get-DtRecommendedRestoreOptions on page 62				X	
Get-DtRepairJobOptionsStatus on page 64				X	
Get-DtScriptCredentials on page 66		X			
Get-DtServerInfo on page 67		X			
Get-DtSnapshot on page 68					X
Get-DtUvraRecommendedFailoverOptions on page 70				X	
Get-DtUvraRecommendedJobOptions on page 72				X	
Get-DtUvraRecommendedRemoveOptions				X	

Cmdlet	Server Object	Server Management	Workload	Job	Engine Control
on page 74					
Get-DtUvraVmHost on page 76				X	
Get-DtVerificationStatus on page 77				X	
Get-DtWorkload on page 78			X		
Get-DtWorkloadPhysicalItem on page 79			X		
Get-DtWorkloadType on page 80			X		
Install-DoubleTake on page 81		X			
Invoke-DtQueueTask on page 84					X
New-DtFilesAndFoldersJob on page 87				X	
New-DtJob on page 89				X	
New-DtServer on page 92	X				
New-DtTaskParameters on page 94					X
New-DtUri on page 95				X	
New-DtUvraServer on page 97	X				
New-DtWorkload on page 99			X		
Remove-DtJob on page 100				X	
Remove-DtPhysicalRule on page 102			X		
Remove-DtSnapshot on page 104					X
Repair-DtJobOptions on page 106				X	
Request-DtOnlineActivation on page 109	X				
Restart-DtReplicationService on page 111		X			
Resume-DtJob on page 112				X	
Resume-DtMirror on page 114					X
Resume-DtTarget on page 116					X
Save-DtJobDiagnostics on page 118				X	

Cmdlet	Server Object	Server Management	Workload	Job	Engine Control
Set-DtActivationCode on page 120		X			
Set-DtBandwidthLimit on page 122					X
Set-DtEmailNotificationOptions on page 124		X			
Set-DtJobCredentials on page 125				X	
Set-DtLogicalItemSelection on page 127			X		
Set-DtOption on page 129		X			
Set-DtPathBlocking on page 131		X			
Set-DtScriptCredentials on page 132		X			
Set-DtServerCredential on page 134	X				
Start-DtJob on page 135				X	
Start-DtJobFailback on page 137				X	
Start-DtJobFailover on page 139				X	
Start-DtJobRestore on page 141				X	
Start-DtJobReverse on page 143				X	
Start-DtMirror on page 145					X
Start-DtOrphansProcessing on page 147					X
Start-DtReplication on page 149					X
Start-DtVerify on page 151					X
Stop-DtJob on page 153				X	
Stop-DtMirror on page 155					X
Stop-DtReplication on page 157					X
Stop-DtReplicationService on page 159		X			
Suspend-DtJob on page 160				X	
Suspend-DtMirror on page 162					X
Suspend-DtTarget on page 164					X

Cmdlet	Server Object	Server Management	Workload	Job	Engine Control
Test-DtActiveDirectoryCredentials on page 166		X			
Test-DtEmailNotification on page 168		X			
Test-DtScript on page 170		X			
Test-DtScriptCredentials on page 172		X			
Undo-DtJobFailover on page 174				X	
Uninstall-DoubleTake on page 176		X			
Update-DtShares on page 177					X
Wait-DtMirrorComplete on page 179					X

Add-DtPhysicalRule

Adds a physical rule to a workload

Syntax

Add-DtPhysicalRule [-ServiceHost] <Server> [-WorkloadId] <Guid> [-Rule] <PhysicalRule> [<CommonParameters>]

Add-DtPhysicalRule [-ServiceHost] <Server> [-WorkloadId] <Guid> -Path <String> [-Recurse] [-Exclude] [<CommonParameters>]

Detailed Description

This cmdlet adds a physical rule to the specified workload on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your source server.	true	false
WorkloadId	Guid on page 258	Specify the workload GUID returned from the New-DtWorkload cmdlet using the workload type name parameter. See New-DtWorkload on page 99.	true	false
Rule	PhysicalRule on page 297	Use the Windows PowerShell New-Object cmdlet to create a physical rule object from DoubleTake.Common.Contract.PhysicalRule.	true	false
Path	String	Specify the path on the source that contains the data that you want to protect	true	false
Recurse	Switch Parameter	Apply the rule to the subdirectories of the specified path. If you do not specify this option, the subdirectories of the specified path will not be included/excluded.	false	false
Exclude	Switch Parameter	Exclude the specified path from mirroring and replication. If you do not specify this option, the path will be included for mirroring and replication.	false	false

Outputs

ChangedItems on page 211

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FilesAndFolders

$DtPhysicalPath = New-Object DoubleTake.Common.Contract.PhysicalRule -Property @ {Path="C:\DirName"}

Add-DtPhysicalRule -ServiceHost $DtServerObjectAlpha -WorkloadId $DtWorkloadGuid -Rule $DtPhysicalPath

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The script then creates a workload on the server for a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. A new object is created from Double-Take.Common.Contract.PhysicalRule to store the physical path C:\DirName in the variable DtPhysicalPath. Finally, the physical rule is added to the workload on the server. The connections for the server object are then closed.

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FilesAndFolders

Add-DtPhysicalRule -ServiceHost $DtServerObjectAlpha -WorkloadId $DtWorkloadGuid -Path "C:\DirName"

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The script then creates a workload on the server for a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. A physical rule is then created for the path C:\DirName. The connections for the server object are then closed.

Add-DtUvraPhysicalRule

Adds a physical rule to a workload

Syntax

Add-DtUvraPhysicalRule [-ServiceHost] <Server> [-Workload] <Workload> -Path <String> [-Recurse] [-Exclude] [<CommonParameters>]

Add-DtUvraPhysicalRule [-ServiceHost] <Server> [-Workload] <Workload> [-Rule] <PhysicalRule> [<CommonParameters>]

Detailed Description

This cmdlet adds a physical rule to the specified full server to ESX appliance workload on the specified server

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtUvraServer cmdlet. See New-DtUvraServer on page 97. For this cmdlet, the -ServiceHost should be your source server.	true	false
Workload	Workload on page 373	Specify the workload object returned from Get-DtUvraRecommendedJobOptions.JobOptions.Workload. See Get-DtUvraRecommendedJobOptions on page 72.	true	false
Path	String	Specify the path on the source that contains the data that you want to protect	true	false
Recurse	Switch Parameter	Apply the rule to the subdirectories of the specified path. If you do not specify this option, the subdirectories of the specified path will not be included/excluded.	false	false
Exclude	Switch Parameter	Exclude the specified path from mirroring and replication. If you do not specify this option, the path will be included for mirroring and replication.	false	false
Rule	PhysicalRule on page 297	Use the Windows PowerShell New-Object cmdlet to create a physical rule object from DoubleTake.Common.Contract.PhysicalRule.	true	false

Outputs

Workload on page 373

Examples

```
$DtServerObjectAlpha= New-DtUvraServer -Name alpha -UserName domain\administrator -Password password -Port 6325

$DtApplianceObject = New-DtUvraServer -Name beta -UserName root -Password password -Port 6325

$DtApplianceHost = New-DtUvraServer -Name gamma -UserName root -Password password

$DtProxyInfo = Get-DtProxiedServerInfo -ServiceHost $DtApplianceObject -Source $DtServerObjectAlpha

$DtVmHostInfo = Get-DtUvraVmHost -ServiceHost $DtApplianceObject -EsxHost $DtApplianceHost -VCenter

$DtRecommendedJobOptions = Get-DtUvraRecommendedJobOptions -ServiceHost $DtApplianceObject -Source $DtServerObjectAlpha -SourceInfo $DtProxyInfo -ApplianceHost $DtVmHostInfo

Add-DtUvraPhysicalRule -ServiceHost $DtApplianceObject -Workload $DtRecommendedJobOptions.JobOptions.Workload -Path "C:\Documents and Settings"

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtApplianceObject

Disconnect-DtServer -ServiceHost $DtApplianceHost
```

Three server objects are created for the source, the appliance, and the ESX server hosting the appliance, assigning the server objects to the DtServerObjectAlpha, DtApplianceObject, and DtApplianceHost variables, respectively. Then proxy and host information is retrieved for those server objects, storing the information in DtProxyInfo and DtVmHostInfo, respectively. That information is then used to retrieve the recommended job options. A rule for the path C:\Documents and Settings is added to the recommended job options. The connections for the server object are then closed.

Checkpoint-DtConnection

Creates a snapshot

Syntax

Checkpoint-DtConnection [-ServiceHost] <Server> [-JobId] <Guid> [-ConnectionId <Guid>] [<CommonParameters>]

Checkpoint-DtConnection [-ServiceHost] <Server> [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet creates a snapshot of the source replica data on the target.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

None

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}
```

```
Checkpoint-DtConnection -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha
Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Then a snapshot of the replica data on the target is taken. The connections for the server object are then closed.

Close-DtWorkload

Closes the workload

Syntax

Close-DtWorkload [-ServiceHost] <Server> [-WorkloadId] <Guid> [<CommonParameters>]

Detailed Description

This cmdlet closes the workload creation process on the specified server and removes all resources associated with the workload creation process.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your source server.	true	false
WorkloadId	Guid on page 258	Specify the workload GUID returned from the New-DtWorkload cmdlet using the workload type name parameter. See New-DtWorkload on page 99.	true	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FilesAndFolders

Close-DtWorkload -ServiceHost $DtServerObjectAlpha -WorkloadId $DtWorkloadGuid

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The script then creates a workload on the server for a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. The workload is then closed on the server. The connections for the server object are then closed.

Confirm-DtJobOptions

Validates job options

Syntax

Confirm-DtJobOptions [-ServiceHost] <Server> [-JobId] <Guid> [[-JobOptions] <JobOptions>] [<CommonParameters>]

Confirm-DtJobOptions [-ServiceHost] <Server> -JobInfo <JobInfo> [-JobOptions <JobOptions>] [<CommonParameters>]

Confirm-DtJobOptions [-ServiceHost] <Server> [-Source] <Server> [-JobType] <String> [-JobOptions] <JobOptions> [[-OtherServers] <Server[]>] [<CommonParameters>]

Detailed Description

This cmdlet validates the job options returned from the Get-DtRecommendedJobOptions cmdlet are compatible with the source and target servers you are using. View the details of the validation by using Get-DtVerificationStatus. See Get-DtRecommendedJobOptions on page 58 and Get-DtVerificationStatus on page 77.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobOptions	JobOptions on page 268	Specify the JobOptions returned from the Get-DtRecommendedJobOptions or Get-DtUvraRecommendedJobOptions cmdlet. See Get-DtRecommendedJobOptions on page 58 or Get-DtUvraRecommendedJobOptions on page 72.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	false
Source	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92.	true	false
JobType	String	Specify the type of job from the following list.	true	false

Name	Type	Description	Required	Pipeline Input
		• **ClusterAwareDthv**—Cluster-aware agentless Hyper-V job • **ClusterAwareFilesAndFolders**—Cluster-aware files and folders job • **ClusterAwareHV2V**—Cluster-aware virtual to Hyper-V job • **ClusterAwareMultiSelectDthv**—Cluster-aware agentless Hyper-V job • **Diagnostics**—Throughput Diagnostic Utility job • **Dthv**—Agentless Hyper-V job • **Exchange**—Exchange job • **ExchangeClustered**—Cluster-aware Exchange job • **FilesAndFolders**—Files and folders job • **FullServerFailover**—Full server job • **Legacy**—Double-Take version 5.3 connections, Double-Take 6.0 connections made outside of the Double-Take Console or Double-Take PowerShell, or GeoCluster Replicated Disk Resource generated jobs • **MoveDataOnlyMigration**—Data migration job • **MoveServerMigration**—Full server migration job • **MultiSelectDthv**—Agentless Hyper-V job • **OrphanedConnection**—Orphaned job • **Sql**—SQL job • **SqlClustered**—Cluster-aware SQL job • **Uvra**—Full server to ESX appliance job • **V2V**—V to ESX or V to Hyper-V job • **Vra**—Full server to ESX or full server to Hyper-V job • **VraMove**—Full server to ESX migration or full server to Hyper-V migration job		
Other Servers	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. Specify multiple server objects in an array using the format @($server1, $server2).	false	false

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

$DtValidation = Confirm-DtJobOptions -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -
JobOptions $DtJob.Options

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The job options used by the job are confirmed, and the validation result is stored in DtValidation. The connections for the server object are then closed.

Disconnect-DtServer

Closes WCF connections

Syntax

Disconnect-DtServer [-ServiceHost] <Server> [<CommonParameters>]

Detailed description

This cmdlet closes all WCF (Windows Communication Foundation) connections that have been opened during use of the server object.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The connections for the server object are then closed.

Edit-DtJob

Edits a job

Syntax

Edit-DtJob [-ServiceHost] <Server> [-JobId] <Guid> [-JobOptions] <JobOptions> [<CommonParameters>]

Edit-DtJob [-ServiceHost] <Server> [-JobOptions] <JobOptions> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet allows you to edit an existing job that is stopped or running, using a JobOptions object that has been modified with your edited job settings.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobOptions	JobOptions on page 268	Specify the JobOptions returned from the Get-DtRecommendedJobOptions or Get-DtUvraRecommendedJobOptions cmdlet. See Get-DtRecommendedJobOptions on page 58 or Get-DtUvraRecommendedJobOptions on page 72.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. See Get-DtJob on page 39. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet.Specify multiple job information objects in an array using the format @($JobInfo1, $JobInfo2).	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
```

```
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

$DtJobForAlpha.Options.CoreMonitorOptions.TotalTimeAllowed="00:10:00"

Edit-DtJob -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -JobOptions
$DtJobForAlpha.Options

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. A job option is changed. In this case, the total time before failover is triggered is set to 10 minutes. Finally, the job options are used to edit the specified job. The connections for the server object are then closed.

Get-DtAccessLevel

Returns the security access level

Syntax

Get-DtAccessLevel [-ServiceHost] <Server> [<CommonParameters>]

Detailed Description

This cmdlet returns the Double-Take security access level for the credentials stored in the specified server object.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false

Outputs

AccessLevel on page 186

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Get-DtAccessLevel -ServiceHost $DtServerObjectAlpha
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the Double-Take security access level for that server is returned. The connections for the server object are then closed.

Get-DtActivationStatus

Returns activation code validation information

Syntax

Get-DtActivationStatus [-ServiceHost] <Server> [[-Code] <String[]>] [[-AdditionalCode] <String[]>] [<CommonParameters>]

Detailed Description

This cmdlet returns the Double-Take activation code validation information for the specified server. If you do not provide the code parameters, the activation codes currently in use will be returned. Specifying the codes will return what the activation status would be if the codes were applied using Set-DtActivationCode on page 120.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Code	String	Specify the 24-character, alpha-numeric activation code (s) which applies the appropriate Double-Take license to your Double-Take server. Specify multiple codes in an array using the format @(code1, code2).	false	false
Additional Code	String	Specify any additional activation codes, such as activation keys. Specify multiple codes in an array using the format @(code1, code2).	false	false

Outputs

ActivationStatus on page 191

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Get-DtActivationStatus -ServiceHost $DtServerObjectAlpha
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the validation information for the Double-Take activation code assigned to the server is returned. The connections for the server object are then closed.

Get-DtBandwidthLimit

Returns bandwidth limiting configuration

Syntax

Get-DtBandwidthLimit [-ServiceHost] <Server> [-JobId] <Guid> [-ConnectionId <Guid>] [<CommonParameters>]

Get-DtBandwidthLimit [-ServiceHost] <Server> [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet returns the bandwidth limiting configuration for the specified job .

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

BandwidthLimit on page 203

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}
```

```
Get-DtBandwidthLimit -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The bandwidth limiting configuration is then returned. The connections for the server object are then closed.

Get-DtConnectionIds

Returns connection ID

Syntax

Get-DtConnectionIds [-ServiceHost] <Server> [[-JobId] <Guid>] [<CommonParameters>]

Detailed Description

This cmdlet returns the connection ID associated with the specified job .

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false

Outputs

ConnectionId on page 216

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

$DtConIdForAlpha = Get-DtConnectionIds -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The connection ID for the job is then stored in DtConIdForAlpha. The connections for the server object are then closed.

Get-DtDiagnostics

Collects support diagnostics

Syntax

Get-DtDiagnostics [-ServiceHost] <Server> [-OutputDirectory] <String> [<CommonParameters>]

Detailed Description

This cmdlet collects configuration data for use when reporting problems to technical support. Because the diagnostics are gathering several pieces of information, potentially across the network to the machine where you are running the cmdlet, it may take several minutes to complete the information gathering and sending the resulting zip file to the cmdlet machine.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Output Directory	String	Specify the full location and file name with a .zip extension, on the machine where you are running the Get-DtDiagnostics cmdlet, to store the resulting zip file containing the diagnostics information.	true	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Get-DtDiagnostics -ServiceHost $DtServerObjectAlpha "C:\AlphaDiagnostics.zip"
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then diagnostics are collected for the server and stored at the root of the C: drive in the file called AlphaDiagnostics.zip. The connections for the server object are then closed.

Get-DtEmailNotificationOptions

Returns e-mail notification settings

Syntax

Get-DtEmailNotificationOptions [-ServiceHost] <Server> [<CommonParameters>]

Detailed Description

This cmdlet returns the current e-mail notification settings for the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false

Outputs

EmailNotificationOptions on page 235

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Get-DtEmailNotificationOptions -ServiceHost $DtServerObjectAlpha
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the current e-mail notification settings for that server are returned. The connections for the server object are then closed.

Get-DtEventLogEntry

Returns a list of event log entries.

Syntax

Get-DtEventLogEntry [-ServiceHost] <Server> [-LastIndex <Int32>] [-ChunkSize <Int32>] [<CommonParameters>]

Detailed Description

This cmdlet returns a list of Double-Take event log entries for the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
LastIndex	Int32	Specify an index entry. The next index entry after the number you specify will be the starting point for the log entries returned. For example, if you specify 144 then the first log entry retrieved will be index 145.	false	false
ChunkSize	Int32	Specify the number of entries that will be returned at one time.	false	false

Outputs

EventLogEntry on page 238

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Get-DtEventLogEntry -ServiceHost $DtServerObjectAlpha -ChunkSize 25
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the Double-Take event entries are displayed, in groups of 25. The connections for the server object are then closed.

Get-DtJob

Returns job information and status for the specified job on the specified server

Syntax

Get-DtJob [-ServiceHost] <Server> [[-JobId] <Guid>] [<CommonParameters>]

Detailed Description

Returns job information and status for the specified job on the specified server. To change the options of an existing job, use Edit-DtJob on page 29.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet. See New-DtJob on page 89. Specify multiple GUID objects in an array using the format @($JobGuid1, $JobGuid2).	false	false

Outputs

JobInfo [ ] on page 266

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FilesAndFolders

$DtWorkload = Get-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadId $DtWorkloadGuid

$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtServerObjectBeta -Source $DtServerObjectAlpha -JobType FilesAndFolders -Workload $DtWorkload

$DtFnFJobGuid = New-DtJob -ServiceHost $DtServerObjectBeta -Source $DtServerObjectAlpha -JobType FilesAndFolders -JobOptions $DtJobOptions.JobOptions

$DtJobInfo = Get-DtJob -ServiceHost $DtServerObjectBeta -JobId $DtFnFJobGuid

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The script then creates a workload on the server for a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. The workload definition for the workload type and the server is then stored in the DtWorkload variable. The recommended job options for the servers and the workload type are then stored in the variable DtJobOptions. A new files and folders job is created using the servers and the job options. The job ID is stored in the variable DtFnFJobGuid. Finally, the job information for the job is stored in the variable DtJobInfo. The connections for the server object are then closed.

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobInfo = Get-DtJob -ServiceHost $DtServerObjectBeta

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. All job information for all of the jobs on the server beta are stored in the variable DtJobInfo. This type of usage is common when the jobs were created in the past or if you did not store or do not know a job's ID. The connections for the server object are then closed.

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. This usage is common for servers that have more than one job, but you only want job information for one specific job. The connections for the server object are then closed.

Get-DtJobActionStatus

Returns the status of a job action

Syntax

Get-DtJobActionStatus [-ServiceHost] <Server> [[-JobId] <Guid>] [<CommonParameters>]

Get-DtJobActionStatus [-ServiceHost] <Server> [-Action] <ActivityToken> [<CommonParameters>]

Detailed Description

This cmdlet returns the status of a job action that has been queued for job. The first syntax returns the status of all of the actions queued for the specified job. The second syntax returns the status for the job action object specified.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	false	false
Action	ActivityToken on page 195	Specify a Double-Take job action object.	true	false

Outputs

ActivityStatusEntry on page 194

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password
$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}
Get-DtJobActionStatus -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id
Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. A server object is created for the server beta using the domain\administrator and

password credentials. It assigns the server object to the variable called DtServerObjectBeta. Finally, the status of the job action is returned. The connections for the server object are then closed.

Get-DtLogicalItem

Returns workload logical items

Syntax

Get-DtLogicalItem [-ServiceHost] <Server> [-WorkloadId] <Guid> [-RefItem <LogicalIteget-gm>] [<CommonParameters>]

Detailed Description

This cmdlet returns the logical items associated with the specified workload.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your source server.	true	false
WorkloadId	Guid on page 258	Specify the workload GUID returned from the New-DtWorkload cmdlet using the workload type name parameter. See New-DtWorkload on page 99.	true	false
RefItem	LogicalItems on page 273	Specify an object returned from a previous Get-DtLogicalItem call.	false	false

Outputs

LogicalItems [ ] on page 273

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FullServerFailover

$DtLogicalItems = Get-DtLogicalItem -ServiceHost $DtServerObjectAlpha -WorkloadId $DtWorkloadGUID

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The script then creates a workload on the server for a full sever job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. The logical items associated with the workload type and the server are then stored in the variable DtLogicalItems. The connections for the server object are then closed.

Get-DtLogMessage

Returns log messages

Syntax

Get-DtLogMessage [-ServiceHost] <Server> [-Source <String>] [-LastSequenceNumber <Int32>] [-LastTimeStamp <DateTimeOffset>] [-ChunkSize <Int32>] [<CommonParameters>]

Detailed Description

This cmdlet returns messages from the Double-Take log file and the Double-Take Management Service log file.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Source	String	Specify source of the log message. If no source is specified, both Double-Take Management Service messages and Double-Take service messages will be returned. To have only Double-Take Management Service messages displayed, specify MS for the string value. To have only Double-Take service messages displayed, specify EN for the string value.	false	false
Last Sequenece Number	Int32	Specify a sequence number to be the starting point to retrieve the log messages.	false	false
LastTime Stamp	DateTime Offset	Specify a date and time stamp to be the starting point to retrieve the log messages. Specify the date in mm/dd/yyyy format. Specify the time in hh:mm:ss format with AM or PM. You can specify a time zone offset, for example, -04:00. If you do not specify a time zone offset, the time zone of the machine you are running from will be used. If you do not specify a time, 12:00:00 AM will be used.	false	false
ChunkSize	Int32	Specify the number of entries that will be returned at one time.	false	false

Outputs

LogMessage [ ] on page 276

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Get-DtLogMessage -ServiceHost $DtServerObjectAlpha -source MS -LastTimeStamp "04/29/2013 05:19:00 PM" -ChunkSize 25

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the Double-Take Management Service log messages after 5:19pm on April 29, 2013 are displayed in groups of 25. The connections for the server object are then closed.

Get-DtOnlineActivationRequest

Returns the server information that is required to activate a license

Syntax

Get-DtOnlineActivationRequest [-ServiceHost] <Server> [<CommonParameters>]

Detailed Description

This cmdlet returns server information, unique to a specific, single server, that is required to activate a Double-Take license. The server must already have an activation code on the server in order to get the server information.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false

Outputs

ServerActivationInformation on page 323

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Get-DtOnlineActivationRequest -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The server information for the online activation process is returned. The connections for the server object are then closed.

Get-DtOption

Returns job or server value

Syntax

Get-DtOption [-ServiceHost] <Server> [[-Name] <String[]>] [<CommonParameters>]

Detailed Description

This cmdlet returns the value of the specific job or server option from the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Name	String	Specify the name of the job or server option. See Server and job settings on page 421 for details on each job and server option. Specify multiple strings in an array using the format @(string1, string2).	false	false

Outputs

Hashtable

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Get-DtOption -ServiceHost $DtServerObjectAlpha -Name MirrorChunkSize
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the value of the server option called MirrorChunkSize is returned from the server. The connections for the server object are then closed.

Get-DtPathBlocking

Returns the blocked paths

Syntax

Get-DtPathBlocking [-ServiceHost] <Server> [<CommonParameters>]

Detailed Description

This cmdlet returns the paths that are blocked on the specified target server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false

Outputs

PathBlocking on page 294

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Get-DtPathBlocking -ServiceHost $DtServerObjectAlpha
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the paths that are blocked on the server are returned. The connections for the server object are then closed.

Get-DtPhysicalItem

Returns files system information

Syntax

This cmdlet returns file system information for the specified server. A physical item can be used to specify a specific file, folder, or volume to return file system information for.

Detailed Description

Get-DtPhysicalItem [-ServiceHost] <Server> [-Ref <PhysicalItem>] [<CommonParameters>]

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Ref	PhysicalItem on page 296	Specify an object returned from a previous Get-DtPhysicalItem call.	false	false

Outputs

PhysicalItem [ ] on page 296

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtVolumes = Get-DtPhysicalItem -ServiceHost $DtServerObjectAlpha

$DtVolume1Root = Get-DtPhysicalItem -ServiceHost $DtServerObjectAlpha -Ref $DtVolumes[0]

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the volumes on the server are stored in the variable DtVolumes. Finally, the files and folders at the root of the first volume in DtVolumes is stored in the variable DtVolume1Root. The connections for the server object are then closed.

Get-DtProductInfo

Returns Double-Take product information

Syntax

Get-DtProductInfo [-ServiceHost] <Server> [<CommonParameters>]

Detailed Description

This cmdlet returns Double-Take product information for the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false

Outputs

ProductInfo on page 300

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Get-DtProductInfo -ServiceHost $DtServerObjectAlpha
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the Double-Take product information for the server is returned. The connections for the server object are then closed.

Get-DtProxiedServerInfo

Returns proxy server information

Syntax

Get-DtProxiedServerInfo [-ServiceHost] <Server> [-Source] <Server> [<CommonParameters>]

Detailed Description

This cmdlet returns proxy server information for the specified server. This information is used in full server to ESX appliance job cmdlets.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtUvraServer cmdlet. See New-DtUvraServer on page 97. For this cmdlet, the -ServiceHost should be your target server.	true	false
Source	Server on page 322	Specify the server object returned from the New-DtUvraServer cmdlet. See New-DtUvraServer on page 97.	true	false

Outputs

ServerInfo on page 324

Examples

```
$DtServerObjectAlpha= New-DtUvraServer -Name alpha -UserName domain\administrator -Password password -Port 6325

$DtApplianceObject = New-DtUvraServer -Name beta -UserName root -Password password -Port 6325

Get-DtProxiedServerInfo -ServiceHost $DtApplianceObject -Source $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtApplianceObject
```

Two server objects are created for the source and the appliance, assigning the server objects to the DtServerObjectAlpha and DtApplianceObject variables, respectively. Then proxy server information is retrieved for those server objects. The connections for the server object are then closed.

Get-DtQualificationResults

Returns the qualification results

Syntax

Get-DtQualificationResults [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Get-DtQualificationResults [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet returns the qualification results for the specified job type. You may want to use these results for job options when editing a job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	false

Outputs

JobQualificationResults on page 269

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Get-DtQualificationResults -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id
Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable

DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The qualification results for the job are then returned. The connections for the server object are then closed.

Get-DtRecommendedFailbackOptions

Returns the recommended failback options

Syntax

Get-DtRecommendedFailbackOptions [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Get-DtRecommendedFailbackOptions [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet returns the recommended failback options for the specified job on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	false

Outputs

RecommendedFailbackOptions on page 303

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password
$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}
Get-DtRecommendedFailbackOptions -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id
Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Then the

recommended failback options for the specified job and server are returned. The connections for the server object are then closed.

Get-DtRecommendedFailoverOptions

Returns the recommended failover options

Syntax

Get-DtRecommendedFailoverOptions [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Get-DtRecommendedFailoverOptions [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters

Detailed Description

This cmdlet returns the recommended failover options for the specified job on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	false	false

Outputs

RecommendedFailoverOptions on page 304

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Get-DtRecommendedFailoverOptions -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id
Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Then the

recommended failover options for the specified job and server are returned. The connections for the server object are then closed.

Get-DtRecommendedJobOptions

Returns recommended job options

Syntax

Get-DtRecommendedJobOptions [-ServiceHost] <Server> [-Source] <Server> [-JobType] <String> [-Workload] <Workload> [-OtherServers <Server[]>] [<CommonParameters>]

Detailed Description

This cmdlet returns the recommended job options for the specified job type.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
Source	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92.	true	false
JobType	String	Specify the type of job from the following list. • **ClusterAwareDthv**—Cluster-aware agentless Hyper-V job • **ClusterAwareFilesAndFolders**—Cluster-aware files and folders job • **ClusterAwareHV2V**—Cluster-aware virtual to Hyper-V job • **ClusterAwareMultiSelectDthv**—Cluster-aware agentless Hyper-V job • **Diagnostics**—Throughput Diagnostic Utility job • **Dthv**—Agentless Hyper-V job • **Exchange**—Exchange job • **ExchangeClustered**—Cluster-aware Exchange job • **FilesAndFolders**—Files and folders job • **FullServerFailover**—Full server job • **Legacy**—Double-Take version 5.3 connections, Double-Take 6.0 connections made outside of the Double-Take Console or Double-Take PowerShell, or GeoCluster Replicated Disk Resource generated jobs • **MoveDataOnlyMigration**—Data migration job • **MoveServerMigration**—Full server migration job	true	false

Name	Type	Description	Required	Pipeline Input
		• **MultiSelectDthv**—Agentless Hyper-V job • **OrphanedConnection**—Orphaned job • **Sql**—SQL job • **SqlClustered**—Cluster-aware SQL job • **Uvra**—Full server to ESX appliance job • **V2V**—V to ESX or V to Hyper-V job • **Vra**—Full server to ESX or full server to Hyper-V job • **VraMove**—Full server to ESX migration or full server to Hyper-V migration job		
Workload	Workload on page 373	Specify the workload object returned from the Get-DtWorkload cmdlet. See Get-DtWorkload on page 78.	true	false
Other Servers	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. Specify multiple server objects in an array using the format @($server1, $server2).	false	false

Outputs

RecommendedJobOptions on page 306

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName
FilesAndFolders

$DtWorkload = Get-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadId $DtWorkloadGuid

$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtServerObjectBeta -Source
$DtServerObjectAlpha -JobType FilesAndFolders -Workload $DtWorkload

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The script then creates a workload on the server for a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. The workload definition for the workload type and the server is then stored in the DtWorkload variable. The recommended job options for the

servers and the workload type are then stored in the variable DtJobOptions. The connections for the server object are then closed.

Get-DtRecommendedPathTransform

Returns the recommended mappings

Syntax

This cmdlet returns the recommended mapping between the location of the data on the source and the location of the replica data on the target for the specific type of workload .

Detailed Description

Get-DtRecommendedPathTransform [-ServiceHost] <Server> [-WorkloadId] <Guid> [-BasePath <String>] [<CommonParameters>]

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your source server.	true	false
WorkloadId	Guid on page 258	Specify the workload GUID returned from the New-DtWorkload cmdlet using the workload type name parameter. See New-DtWorkload on page 99.	true	false
BasePath	String	Specify the location on the target where the replica of the source data will be stored. By default, the replica source data will be stored in the same directory structure on the target, in a one-to-one configuration.	false	false

Outputs

PathTransformation [ ] on page 295

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FullServerFailover

Get-DtRecommendedPathTransform -ServiceHost $DtServerObjectAlpha -WorkloadId $DtWorkloadGuid

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The script then creates a workload on the server for a full sever job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. The recommended mapping of the data on the source and the replica data on the target is then returned. The connections for the server object are then closed.

Get-DtRecommendedRestoreOptions

Returns the recommended restoration options

Syntax

Get-DtRecommendedRestoreOptions [-ServiceHost] <Server> [-JobId] <Guid> [-RestoreTarget <Server>] [-RequestCanClearRestoreRequired] [<CommonParameters>]

Get-DtRecommendedRestoreOptions [-ServiceHost] <Server> [-RestoreTarget <Server>] [-RequestCanClearRestoreRequired] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

Returns the recommended restoration options for the specified job on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Restore Target	Server on page 322	Specify the server you wan to restore to. If you do not specify a server, the original source server will be used.	false	false
Request CanClear Restore Required	Switch Parameter	Clears the restore required state of the job	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	false

Outputs

RecommendedRestoreOptions on page 305

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
```

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Get-DtRecommendedRestoreOptions -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -
RestoreTarget $DtServerObjectAlpha -RequestCanClearRestoreRequired

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Then the recommended restore options for the specified job and server are returned. The connections for the server object are then closed.

Get-DtRepairJobOptionsStatus

Returns the details and status of a repair

Syntax

Get-DtRepairJobOptionsStatus [-ServiceHost] <Server> [-Token] <ActivityToken> [<CommonParameters>]

Detailed Description

This cmdlet returns the details and status of the repair performed by the Repair-DtJobOptions cmdlet. See Repair-DtJobOptions on page 106.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
Token	ActivityToken on page 195	Specify the repair action object returned from the Repair-DtJobOptions cmdlet. See Repair-DtJobOptions on page 106.	true	false

Outputs

RepairStatus on page 308

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

$DtValidation = Confirm-DtJobOptions -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -
JobOptions $DtJob.Options

$DtStatus = Get-DtVerificationStatus -ServiceHost $DtServerObjectBeta -Token $DtValidation

$DtRepair = Repair-DtJobOptions -ServiceHost $DtTarget -JobId $DtJob.Id -JobOptions $DtJob.Options -
Step $DtStatus.Steps

$DtRepairStatus = Get-DtRepairJobOptionsStatus -ServiceHost $DtServerObjectBeta -Token $DtRepair

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The job options

used by the job are confirmed, and the validation result is stored in DtValidation. The details of the validation are stored in the variable DtStatus. Those items that can automatically be fixed are corrected. If the job options were modified in order to fix an issues, the updated job options are now contained in the variable $DtRepair. The details and status of the repair are stored in the variable DtRepairStatus. The connections for the server object are then closed.

Get-DtScriptCredentials

Returns credentials

Syntax

Get-DtScriptCredentials [-ServiceHost] <Server> [<CommonParameters>]

Detailed Description

This cmdlet returns the credentials that Double-Take is currently using to run scripts on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Get-DtScriptCredentials -ServiceHost $DtServerObjectAlpha
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the credentials that Double-Take is using to run scripts on this server are returned. The connections for the server object are then closed.

Get-DtServerInfo

Returns server information

Syntax

Get-DtServerInfo [-ServiceHost] <Server> [<CommonParameters>]

Detailed Description

This cmdlet returns server configuration information for the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false

Outputs

ServerInfo on page 324

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Get-DtServerInfo -ServiceHost $DtServerObjectAlpha
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then server configuration information for the server is returned. The connections for the server object are then closed.

Get-DtSnapshot

Returns snapshots

Syntax

Get-DtSnapshot [-ServiceHost] <Server> [-JobId] <Guid> [-ConnectionId<Guid>] [<CommonParameters>]

Get-DtSnapshot [-ServiceHost] <Server> [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet returns the available Double-Take snapshots for the specified job .

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

SnapshotEntry on page 333

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Get-DtSnapshot -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The snapshots available for the job are returned. The connections for the server object are then closed.

Get-DtUvraRecommendedFailoverOptions

Returns the recommended failover options

Syntax

Get-DtUvraRecommendedFailoverOptions [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Get-DtUvraRecommendedFailoverOptions [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet returns the recommended failover options for the specified full server to ESX appliance job on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtUvraServer cmdlet. See New-DtUvraServer on page 97. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	false	false

Outputs

FailoverOptions on page 246

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName root -Password password

$DtApplianceObject = New-DtUvraServer -Name beta -UserName root -Password password -Port 6325

$DtJobForAlpha = Get-DtJob -ServiceHost $DtApplianceObject | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Get-DtUvraRecommendedFailoverOptions -ServiceHost $DtApplianceObject -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtApplianceObject
```

```
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the root and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. A server object is created for the appliance beta using port 6325 and the root and password credentials. It assigns the server object to the variable called DtApplianceObject. The job (s) are retrieved from DtApplianceObject, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Then the recommended failover options for the specified full server to ESX appliance job and server are returned. The connections for the server object are then closed.

Get-DtUvraRecommendedJobOptions

Returns the recommended job options

Syntax

Get-DtUvraRecommendedJobOptions [-ServiceHost] <Server> [-Source] <Server> [-SourceInfo] <ServerInfo> [-TargetInfo] <ServerInfo> [-ApplianceHost] <VmHost> [-OtherServers <Server[]>] [<CommonParameters>]

Detailed Description

This cmdlet returns the recommended job options for the specified full server to ESX appliance job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtUvraServer cmdlet. See New-DtUvraServer on page 97. For this cmdlet, the -ServiceHost should be your target server.	true	false
Source	Server on page 322	Specify the server object returned from the New-DtUvraServer cmdlet. See New-DtUvraServer on page 97.	true	false
SourceInfo	ServerInfo on page 324	Specify the server object returned from the Get-DtProxiedServerInfo cmdlet. See Get-DtProxiedServerInfo on page 51.	true	false
TargetInfo	ServerInfo on page 324	Specify the server object returned from the Get-DtServerInfo cmdlet. See Get-DtServerInfo on page 67.	true	false
Appliance Host	VmHost on page 362	Specify the host object returned from the Get-DtUvraVmHost cmdlet. See Get-DtUvraVmHost on page 76.	true	false
Other Servers	Server on page 322	Specify the server object returned from the New-DtUvraServer cmdlet. See New-DtUvraServer on page 97. Specify the server object in an array using the format @($server).	false	false

Outputs

RecommendedJobOptions on page 306

Examples

```
$DtServerObjectAlpha= New-DtUvraServer -Name alpha -UserName domain\administrator -Password
```

```
password -Port 6325

$DtApplianceObject = New-DtUvraServer -Name beta -UserName root -Password password -Port 6325

$DtApplianceHost = New-DtUvraServer -Name gamma -UserName root -Password password

$DtProxyInfo = Get-DtProxiedServerInfo -ServiceHost $DtApplianceObject -Source $DtServerObject

$DtVmHostInfo = Get-DtUvraVmHost -ServiceHost $DtApplianceObject -EsxHost $DtApplianceHost -
VCenter

$DtApplianceServerInfo = Get-DtServerInfo -ServiceHost $DtAppliance

$DtRecommendedJobOptions = Get-DtUvraRecommendedJobOptions -ServiceHost $DtApplianceObject -
Source $DtServerObjectAlpha -SourceInfo $DtProxyInfo -TargetInfo $DtApplianceServerInfo ApplianceHost
$DtVmHostInfo

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtApplianceObject

Disconnect-DtServer -ServiceHost $DtApplianceHost
```

Three server objects are created for the source, the appliance, and the ESX server hosting the appliance, assigning the server objects to the DtServerObject, DtApplianceObject, and DtApplianceHost variables, respectively. Then proxy and host information is retrieved for those server objects, assigning the information to DtProxyInfo and DtVmHostInfo, respectively. Also, the server information for the appliance is retrieved and stored in DtApplianceServerInfo. That information is then used to retrieve the recommended job options. The connections for the server object are then closed.

Get-DtUvraRecommendedRemoveOptions

Returns recommended removal options

Syntax

Get-DtUvraRecommendedRemoveOptions [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Get-DtUvraRecommendedRemoveOptions [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet returns the recommended removal options when deleting the specified full server to ESX appliance job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtUvraServer cmdlet. See New-DtUvraServer on page 97. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	false	false

Outputs

DeleteOptions on page 227

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName root -Password password

$DtApplianceObject = New-DtUvraServer -Name beta -UserName root -Password password -Port 6325

$DtJobForAlpha = Get-DtJob -ServiceHost $DtApplianceObject | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Get-DtUvraRecommendedRemoveOptions -ServiceHost $DtApplianceObject -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtApplianceObject
```

```
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the root and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. A server object is created for the appliance beta using port 6325 and the root and password credentials. It assigns the server object to the variable called DtApplianceObject. The job (s) are retrieved from DtApplianceObject, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Then the recommended remove options for the specified full server to ESX appliance job and server are returned. The connections for the server object are then closed.

Get-DtUvraVmHost

Returns host information

Syntax

Get-DtUvraVmHost [-ServiceHost] <Server> [-EsxHost] <Server> -VCenter [<CommonParameters>]

Detailed Description

This cmdlet returns host information for the specified VMware host. This information is used in full server to ESX appliance job cmdlets.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtUvraServer cmdlet. See New-DtUvraServer on page 97. For this cmdlet, the -ServiceHost should be your target server.	true	false
EsxHost	Server on page 322	Specify the server object returned from the New-DtUvraServer cmdlet. See New-DtUvraServer on page 97.	true	false
VCenter	Switch Parameter	Specify if you are using vCenter. If you do not specify this option, the server will be considered a stanalone ESX host.	false	false

Outputs

VmHost on page 362

Examples

```
$DtApplianceObject = New-DtUvraServer -Name beta -UserName root -Password password -Port 6325

$DtApplianceHost = New-DtUvraServer -Name gamma -UserName root -Password password

Get-DtUvraVmHost -ServiceHost $DtApplianceObject -EsxHost $DtApplianceHost -VCenter
Disconnect-DtServer -ServiceHost $DtApplianceObject

Disconnect-DtServer -ServiceHost $DtApplianceHost
```

Two server objects are created for the appliance and the ESX server hosting the appliance, assigning the server objects to the DtApplianceObject and DtApplianceHost variables, respectively. Then host information is retrieved for those server objects. The connections for the server object are then closed.

Get-DtVerificationStatus

Returns the validation details and status

Syntax

Get-DtVerificationStatus [-ServiceHost] <Server> [-Token] <ActivityToken> [<CommonParameters>]

Detailed Description

This cmdlet returns the details and status of the validation performed by the Confirm-DtJobOptions cmdlet. See Confirm-DtJobOptions on page 25.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
Token	ActivityToken on page 195	Specify the confirm action object returned from the Confirm-DtJobOption cmdlet. See Confirm-DtJobOptions on page 25.	true	false

Outputs

VerificationStatus on page 351

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

$DtValidation = Confirm-DtJobOptions -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -
JobOptions $DtJob.Options

$DtStatus = Get-DtVerificationStatus -ServiceHost $DtServerObjectBeta -Token $DtValidation

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The job options used by the job are confirmed, and the validation result is stored in DtValidation. The details of the validation are stored in the variable DtStatus. The connections for the server object are then closed.

Get-DtWorkload

Returns the workload definition

Syntax

Get-DtWorkload [-ServiceHost] <Server> [-WorkloadId] <Guid> [<CommonParameters>]

Detailed Description

This cmdlet returns an object that represents the workload definition, including the workload type name, any physical rules, and any logical rules. This object is used in job cmdlets.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your source server.	true	false
WorkloadId	Guid on page 258	Specify the workload GUID returned from the New-DtWorkload cmdlet using the workload type name parameter. See New-DtWorkload on page 99.	true	false

Outputs

Workload on page 373

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FullServerFailover

$DtWorkload = Get-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadId $DtWorkloadGuid

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The script then creates a workload on the server for a full sever job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. The workload definition for the workload type and the server is then stored in the DtWorkload variable. The connections for the server object are then closed.

Get-DtWorkloadPhysicalItem

Returns physical items

Syntax

Get-DtWorkloadPhysicalItem [-ServiceHost] <Server> [-WorkloadId] <Guid> [-RefItem <PhysicalItem>] [<CommonParameters>]

Detailed Description

This cmdlet returns the physical items available for the specified workload on the specified server

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your source server.	true	false
WorkloadId	Guid on page 258	Specify the workload GUID returned from the New-DtWorkload cmdlet using the workload type name parameter. See New-DtWorkload on page 99.	true	false
Ref	PhysicalItem on page 296	Specify an object returned from a previous Get-DtWorkloadPhysicalItem call.	false	false

Outputs

PhysicalItem on page 296

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FilesAndFolders

Get-DtWorkloadPhysicalItem -ServiceHost $DtServerObjectAlpha -WorkloadId $DtWorkloadGuid

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The script then creates a workload on the server for a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. Finally, the physical items available for the workload on the server are returned. The connections for the server object are then closed.

Get-DtWorkloadType

Returns the workload types

Syntax

Get-DtWorkloadType [-ServiceHost] <Server> [<CommonParameters>]

Detailed Description

This cmdlet returns the types of workloads that are supported on the specified server. The supported workload types are based on the Double-Take activation code(s) on the server, the applications on the server, the server configuration (like standalone or cluster), and so on.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your source server.	true	false

Outputs

WorkloadType [ ] on page 375

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Get-DtWorkloadType -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then it returns the workload types that are supported on the server. The connections for the server object are then closed.

Install-DoubleTake

Installs Double-Take

Syntax

Install-DoubleTake [-RemoteServer] <Server> -ActivationCode <String[]> [-PackageBaseFolder <String>] [-X86PackageFolder <String>] [-X64PackageFolder <String>] [-DotNetPackagePath <String>] [-TempFolder <String>] [-InstallationFolder <String>] [-DiskQueueFolder <String>] [-DiskQueueLimit <Int32>] [-MaxMemoryUsage <Int32>] [-MinFreeDiskSpace <Int32>] [-SimultaneousFilePushLimit <Int32>] [-AsJob] [<CommonParameters>]

Install-DoubleTake [-RemoteServer] <Server> -ActivationCode <String[]> -Schedule <DateTime> [-NoReboot] [-PackageBaseFolder <String>] [-X86PackageFolder <String>] [-X64PackageFolder <String>] [-DotNetPackagePath <String>] [-TempFolder <String>] [-InstallationFolder <String>] [-DiskQueueFolder <String>] [-DiskQueueLimit <Int32>] [-MaxMemoryUsage <Int32>] [-MinFreeDiskSpace <Int32>] [-SimultaneousFilePushLimit <Int32>] [-AsJob] [<CommonParameters>]

Detailed Description

This cmdlet installs Double-Take on the specified server. The first syntax allows you to install Double-Take immediately. The second syntax allows you to schedule the installation.

Parameters

Name	Type	Description	Required	Pipeline Input
Remote Server	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92.	true	false
Activation Code	String	Specify your 24-character, alpha-numeric activation code(s) which applies the appropriate Double-Take license to your Double-Take server. Specify multiple codes in an array using the format @(code1, code2).	true	false
Package Base Folder	String	Specifies the locations of the setup files (on the local machine) that will be used to install on both 32-bit and 64-bit servers. By default, these are in the i386\ and \x64 subdirectories where you installed Double-Take.	false	false
X86 Package Folder	String	Specify the location of the setup file (on the local machine) that will be used to install on 32-bit servers. By default, this is in the \i386 subdirectory where you installed Double-Take.	false	false
X64 Package Folder	String	Specify the location of the setup file (on the local machine) that will be used to install on 64-bit servers. By default, this is in the \x64 subdirectory where you installed Double-Take.	false	false

Name	Type	Description	Required	Pipeline Input
DotNet Package Path	String	If your servers are running Windows 2008 or earlier and do not have Microsoft .NET version 3.5.1, specify the location of the setup file (on the local machine) that will be used to install it. The setup file is available on the Double-Take CD in the \NetFx\v3.5SP1\Full directory or from the Microsoft web site.	false	false
Temp Folder	String	Specify a temporary location (on the server where you are installing Double-Take) where the installation files will be copied and run. The default is \Temp. You need approximately 130 MB of space in the specified location.	false	false
Installation Folder	String	Specify the location where you want to install Double-Take on the server. The default is \Program Files\Vision Solutions\Double-Take.	false	false
DiskQueue Folder	String	Specify the location where you want to store the Double-Take disk queue on each server. The default is \Program Files\Vision Solutions\Double-Take.	false	false
DiskQueue Limit	Int32	Specify a fixed amount of disk space, in MB, in the specified DiskQueueFolder that can be used for Double-Take disk queuing. When the disk space limit is reached, Double-Take will automatically begin the auto-disconnect process. By default, an unlimited amount of disk queuing will be allowed.	false	false
Max Memory Usage	Int32	Specify the maximum amount of memory, in MB, that can be used for Double-Take processing. The default will depend on your operating system and hardware. For complete details on memory usage, see the Double-Take *User's Guide*.	false	false
MinFree DiskSpace	Int32	This is the minimum amount of disk space in the specified DiskQueueFolder that must be available at all times. This amount should be less than the amount of physical disk space minus the disk size specified for DiskQueueLimit. The default is 50 MB.	false	false
Simultaneous FilePush Limit	Int32	Specify the number of files that can simultaneously be pushed to the machine you are installing on. The default is 5.	false	false
AsJob	Switch Parameter	Specify if you want the installation to occur asynchronously in the background, returning the PowerShell command immediately. You can get the status of each installation using the Windows PowerShell Get-Job command. Without this parameter,	false	false

Name	Type	Description	Required	Pipeline Input
		each push installation specified will be executed synchronously and the current activity of the current installation will be displayed.		
Schedule	DateTime	Specify a date and time to complete the installation. Specify the date in mm/dd/yyyy format. Specify the time in hh:mm:ss format with AM or PM. You can specify a time zone offset, for example, -04:00. If you do not specify a time zone offset, the time zone of the machine you are running from will be used. If you do not specify a time, 12:00:00 AM will be used.	true	false
NoReboot	Switch Parameter	Specify if you do not want the server to reboot after the installation, even if a reboot is required.	false	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Install-DoubleTake -RemoteServer $DtServerObjectAlpha -ActivationCode 1234567890abcdefghij1234

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then Double-Take is installed to the server using the activation code 1234567890abcdefghij1234. The connections for the server object are then closed.

Invoke-DtQueueTask

Queues tasks

Syntax

Invoke-DtQueueTask [-ServiceHost] <Server> [-JobId] <Guid> [-OnQueue <TaskParameters>] [-OnTransmit <TaskParameters>] [-OnReceive <TaskParameters>] [-OnExecute <TaskParameters>] [-InteractWithDesktop] [-Timeout <TimeSpan>] [-ConnectionId <Guid>] [<CommonParameters>]

Invoke-DtQueueTask [-ServiceHost] <Server> [-OnQueue <TaskParameters>] [-OnTransmit <TaskParameters>] [-OnReceive <TaskParameters>] [-OnExecute <TaskParameters>] [-InteractWithDesktop] [-Timeout <TimeSpan>] [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet queues tasks inline with replication data. Keep the following in mind when using this cmdlet.

- Any combination of one or more execution points can be used with the same Invoke-DtQueueTask cmdlet.
- All script processing messages, including errors, can be viewed in the Double-Take log and the Windows Event log.
- If your source is in a restore required state (after a failover), any task placed on the queue will be executed immediately. Use caution when submitting tasks while in this state so that the target does not get inadvertently updated.
- If a task is submitted after replication is stopped, the task will be executed immediately.
- A task may be discarded if all jobs to a target are manually stopped, if replication is stopped to a target, or if an auto-disconnect occurs.
- If you disable task command processing while tasks are in queue, those tasks will not be executed.
- The user submitting the task command must be a member of the Double-Take Admin security group on both the source and target and the Double-Take service must have proper privileges to access the files or run the commands specified in the task.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
OnQueue	TaskParameters on page 345	Execute the specified task on the source machine as soon as the source receives and queues the	false	false

Name	Type	Description	Required	Pipeline Input
		task. During heavy replication, there may be a delay while the task is queued inline with the replication operations. Define the task parameters by using New-DtTaskParameters on page 94.		
OnTransmit	TaskParameters on page 345	Execute the specified task on the source machine just before the source transmits the task to the target. Define the task parameters by using New-DtTaskParameters on page 94.	false	false
OnReceive	TaskParameters on page 345	Execute the specified task on the target machine as soon as the target receives and queues the task. Define the task parameters by using New-DtTaskParameters on page 94.	false	false
OnExecute	TaskParameters on page 345	Execute the specified task on the target when the target processes the task from the queue. Since the task is not executed until it is processed, if the target is paused, the task will be held in queue. Define the task parameters by using New-DtTaskParameters on page 94.	false	false
Interact With Desktop	SwitchParameter	Tasks interact with the desktop and, therefore, display on screen and run in the foreground. If you do not use this option, tasks do not interact with the desktop and will be run in the background.	false	false
Timeout	TimeSpan	Specify the length of time, in timespan format, to wait for tasks to complete. For example, 0.01:30:00 would wait for one hour and thirty minutes. If you set the timespan to zero (0.00:00:00), there is no timeout delay and the next operation is immediately processed. If you do not specify a timeout parameter, the timeout will default to forever.	false	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

$DtScript = New-DtTaskParameters -ScriptPath "C:\PathDir\ScriptName" -Arguments "arg1 arg2"

$DtPsScript = New-DtTaskParameters -ScriptPath
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -Arguments "-File
""C:\PathDir\Script.ps1"" ""-Arg1 argument1_info -Arg2 argument2_info"" -ExecutionPolicy RemoteSigned"

Invoke-DtQueueTask -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -OnReceive $DtScript -
OnExecute $DtPsScript

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The script called ScriptName, located in C:\PathDir, along with two arguments, is stored in the variable DtScript. The script to launch PowerShell and run the script called Script.ps1, located in C:\PathDir, along with two arguments and the ExecutionPolicy parameter, is stored in the variable DtPsScript. Finally, the script stored in DtScript is executed when the target receives and queues the task and the script stored in DtPsScript is executed when the target processes the task from the queue. The connections for the server object are then closed.

New-DtFilesAndFoldersJob

Creates a files and folders job

Syntax

New-DtFilesAndFoldersJob [-ServiceHost] <Server> [-Source] <Server> [-Path] <String> [[-TargetPath] <String>] [-Name <String>] [-JobOptions <JobOptions>] [<CommonParameters>]

Detailed Description

This cmdlet creates a files and folders job on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
Source	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92.	true	false
Path	String	Specify the path on the source that contains the data that you want to protect	true	false
TargetPath	String	Specify the path on the target where you want to store the replica data from the source. By default, a one-to-one mapping will be used on the target, which means the replica source data will be stored in the same directory structure on the target.	false	false
Name	String	Specify the name of the job.	false	false
JobOptions	JobOptions on page 268	Specify the JobOptions returned from the Get-DtRecommendedJobOptions or Get-DtUvraRecommendedJobOptions cmdlet. See Get-DtRecommendedJobOptions on page 58 or Get-DtUvraRecommendedJobOptions on page 72.	false	false

Outputs

Guid on page 258

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password
```

```
$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName
FilesAndFolders

$DtWorkload = Get-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadId $DtWorkloadGuid

$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtServerObjectBeta -Source
$DtServerObjectAlpha -JobType FilesAndFolders -Workload $DtWorkload

New-DtFilesAndFoldersJob -ServiceHost $DtServerObjectBeta -Source $DtServerObjectAlpha -Path
"C:\Data" -TargetPath "C:\Alpha\C" -Name "Alpha to Beta" -JobOptions $DtJobOptions.JobOptions

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The script then creates a workload on the server for a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. The workload definition for the workload type and the server is then stored in the DtWorkload variable. The recommended job options for the servers and the workload type are then stored in the variable DtJobOptions. A new files and folders job is created using the servers and the job options. The connections for the server object are then closed.

New-DtJob

Creates a job

Syntax

New-DtJob [-ServiceHost] <Server> [-Source] <Server> [-JobType] <String> [-JobOptions] <JobOptions> [[-OtherServers] <Server[]>] [<CommonParameters>]

Detailed Description

This cmdlet creates the specified job type on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
Source	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92.	true	false
JobType	String	Specify the type of job from the following list. • **ClusterAwareDthv**—Cluster-aware agentless Hyper-V job • **ClusterAwareFilesAndFolders**—Cluster-aware files and folders job • **ClusterAwareHV2V**—Cluster-aware virtual to Hyper-V job • **ClusterAwareMultiSelectDthv**—Cluster-aware agentless Hyper-V job • **Diagnostics**—Throughput Diagnostic Utility job • **Dthv**—Agentless Hyper-V job • **Exchange**—Exchange job • **ExchangeClustered**—Cluster-aware Exchange job • **FilesAndFolders**—Files and folders job • **FullServerFailover**—Full server job • **Legacy**—Double-Take version 5.3 connections, Double-Take 6.0 connections made outside of the Double-Take Console or Double-Take PowerShell, or GeoCluster Replicated Disk Resource generated jobs • **MoveDataOnlyMigration**—Data migration job	true	false

Name	Type	Description	Required	Pipeline Input
		• **MoveServerMigration**—Full server migration job • **MultiSelectDthv**—Agentless Hyper-V job • **OrphanedConnection**—Orphaned job • **Sql**—SQL job • **SqlClustered**—Cluster-aware SQL job • **Uvra**—Full server to ESX appliance job • **V2V**—V to ESX or V to Hyper-V job • **Vra**—Full server to ESX or full server to Hyper-V job • **VraMove**—Full server to ESX migration or full server to Hyper-V migration job		
JobOptions	JobOptions on page 268	Specify the JobOptions returned from the Get-DtRecommendedJobOptions or Get-DtUvraRecommendedJobOptions cmdlet. See Get-DtRecommendedJobOptions on page 58 or Get-DtUvraRecommendedJobOptions on page 72.	false	false
Other Servers	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. Specify multiple server objects in an array using the format @($server1, $server2).	false	false

Outputs

Guid on page 258

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FilesAndFolders

$DtWorkload = Get-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadId $DtWorkloadGuid

$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtServerObjectBeta -Source $DtServerObjectAlpha -JobType FilesAndFolders -Workload $DtWorkload

$DtFnFJobGuid = New-DtJob -ServiceHost $DtServerObjectBeta -Source $DtServerObjectAlpha -JobType FilesAndFolders -JobOptions $DtJobOptions.JobOptions

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The script then creates a workload on the server for a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. The workload definition for the workload type and the server is then stored in the DtWorkload variable. The recommended job options for the servers and the workload type are then stored in the variable DtJobOptions. A new files and folders job is created using the servers and the job options. The job ID is stored in the variable DtFnFJobGuid. The connections for the server object are then closed.

New-DtServer

Creates a server object

Syntax

New-DtServer [-Name] <String> [[-UserName] <String>] [[-Password] <String>] [-Role <String>] [<CommonParameters>]

New-DtServer [-Name] <String> -Credential <PSCredential> [-Role <String>] [<CommonParameters>]

Detailed Description

This cmdlet creates a server object with specific credentials associated with it. This may be any type of server in your organization, for example a Double-Take server, a DNS server, an application server, and so on. This object is used to communicate with the Double-Take Management Service. You should close the connections to this server object when you are finished using it by using Disconnect-DtServer on page 28.

Parameters

Name	Type	Description	Required	Pipeline Input
Name	String	Specify the name or IP address of the server, cluster, or cluster node.	true	false
UserName	String	Specify a user name.	true	false
Password	String	Specify the password associated with the user you have entered. This password will be visible in plain text.	true	false
Role	String	Specify one of the following roles for the server object you are creating. These servers are used when you specify the -OtherServers parameter in other cmdlets. • **TargetVimServer**—This is the ESX server or vCenter that will host the replica virtual machine after failover. If you are using vCenter, specify your vCenter. Only specify an ESX host if you are using ESX standalone. • **ReverseVimServer**—This is the ESX server or vCenter server that will host the replica virtual after reverse and failover. If you are using vCenter, specify your vCenter. Only specify an ESX host if you are using ESX standalone. • **ReverseHelperRole**—This is the target where data will be replicated after a reverse.	false	false
Credential	PSCredential on page 302	Specify the credential object returned from the Windows PowerShell Get-Credential cmdlet. This	true	false

Name	Type	Description	Required	Pipeline Input
		password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.		

Outputs

Server on page 322

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The connections for the server object are then closed.

```
$DtCredentialEncrypted = Get-Credential
$DtServerObjectAlpha = New-DtServer -Name alpha -Credential $DtCredential
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

User credentials are stored in a variable called $DtCredential. The script will prompt you to supply the username and password and the credentials will be encrypted. Then the stored credentials are used to create a new server object for the server alpha. It assigns the server object to the variable called DtServerObject. The connections for the server object are then closed.

New-DtTaskParameters

Creates parameter set

Syntax

New-DtTaskParameters [-ScriptPath] <String> [[-Arguments] <String>] [<CommonParameters>]

Detailed Description

Creates a parameter set to be used with the Invoke-DtQueueTask cmdlet. See Invoke-DtQueueTask on page 84.

Parameters

Name	Type	Description	Required	Pipeline Input
ScriptPath	String	Specify the full path and script name	true	false
Arguments	String	Specify any arguments that need to be passed to the script.	false	false

Outputs

TaskParameters on page 345

Examples

```
$DtScript = New-DtTaskParameters -ScriptPath "C:\PathDir\ScriptName" -Arguments "arg1 arg2"
```

The script called ScriptName, located in C:\PathDir, along with two arguments, is stored in the variable DtScript.

```
$DtPsScript = New-DtTaskParameters -ScriptPath
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -Arguments "-File
""C:\PathDir\Script.ps1"" ""-Arg1 argument1_info -Arg2 argument2_info"" -ExecutionPolicy RemoteSigned"
```

The script to launch PowerShell and run the script called Script.ps1, located in C:\PathDir, along with two arguments and the ExecutionPolicy parameter, is stored in the variable DtPsScript.

New-DtUri

Creates a URI

Syntax

New-DtUri [-Literal] <String> [<CommonParameters>]

New-DtUri [-NetworkId] <String> [-Credential <PSCredential>] [-Port <Int32>] [-Scheme <String>] [-Query <String>] [-Fragment <String>] [<CommonParameters>]

Detailed Description

This cmdlet creates a URI (uniform resource identifier) that is used to specify job credentials

Parameters

Name	Type	Description	Required	Pipeline Input
Literal	String	Specify the entire URI string.	true	false
NetworkId	String	Specify the name or IP address of the server.	true	false
Credential	PSCredential on page 302	Specify the credential object returned from the Windows PowerShell Get-Credential cmdlet. This password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.	false	false
Port	Int32	Specify the communications port.	false	false
Scheme	String	Specify the scheme name.	false	false
Query	String	Specify any additional identification information.	false	false
Fragment	String	Specify any identifying information that provides direction to a secondary resource.	false	false

Outputs

Uri

Examples

```
New-DtUri -Literal "http://server:6320"
```

A URI is created for http://server:6320.

```
New-DtUri -Literal
"foo://username:password@domain.com:6320/location/index?type=volume&directory=C#location "
```

A URI is created for
foo://username:password@domain.com:6320/location/index?type=volume&directory=C#location.

New-DtUvraServer

Creates a server object

Syntax

New-DtUvraServer [-Name] <String> [[-UserName] <String>] [[-Password] <String>] [-Port <Int32>] [-Role <String>] [<CommonParameters>]

New-DtUvraServer [-Name] <String> [-Port <Int32>] -Credential <PSCredential> [-Role <String>] [<CommonParameters>]

Detailed Description

This cmdlet creates a server object with specific credentials associated with it. This cmdlet is specific to the full server to ESX appliance job type, The server object may be one of the following types of servers: Double-Take server, virtual recovery appliance, or VMware host. The object is used to communicate with the Double-Take Management Service.

Parameters

Name	Type	Description	Required	Pipeline Input
Name	String	Specify the name or IP address of the server, cluster, or cluster node.	true	false
Username	String	Specify a user name.	true	false
Password	String	Specify the password associated with the user you have entered. This password will be visible in plain text.	true	false
Port	Int32	Specify the port for the XML web service protocol. By default, that is 443. Use 6325 for Double-Take servers and appliances, unless you changed the default Double-Take port. Do not specify a port for VMware hosts.	false	false
Role	String	Specify one of the following roles for the server object you are creating. These servers are used when you specify the -OtherServers parameter in other cmdlets. • **TargetVimServer**—This is the ESX server or vCenter that will host the replica virtual machine after failover. If you are using vCenter, specify your vCenter. Only specify an ESX host if you are using ESX standalone. • **ReverseVimServer**—This is the ESX server or vCenter server that will host the replica virtual after reverse and failover. If you are using vCenter, specify your vCenter. Only specify an ESX host if you are using ESX standalone.	false	false

Name	Type	Description	Required	Pipeline Input
		• **ReverseHelperRole**—This is the target where data will be replicated after a reverse.		
Credential	PSCredential on page 302	Specify the credential object returned from the Windows PowerShell Get-Credential cmdlet. This password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.	true	false

Outputs

Server on page 322

Examples

```
$DtServerObjectAlpha= New-DtUvraServer -Name alpha -UserName domain\administrator -Password password -Port 6325

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using port 6325 and the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObject. The connections for the server object are then closed.

```
$DtCredentialEncrypted = Get-Credential

$DtServerObjectAlpha = New-DtUvraServer -Name alpha -Credential $DtCredential -Port 6325

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

User credentials are stored in a variable called $DtCredential. The script will prompt you to supply the username and password and the credentials will be encrypted. Then the stored credentials are used to create a new server object for the server alpha using port 6325. It assigns the server object to the variable called DtServerObject. The connections for the server object are then closed.

New-DtWorkload

Creates a workload

Syntax

New-DtWorkload [-ServiceHost] <Server> -WorkloadTypeName <String> [<CommonParameters>]

New-DtWorkload [-ServiceHost] <Server> -Workload <Workload> [<CommonParameters>]

Detailed Description

This cmdlet creates a Double-Take workload on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your source server.	true	false
Workload TypeName	String	Specify a supported workload type from the Get-DtWorkloadType cmdlet. See Get-DtWorkloadType on page 80.	true	false
Workload	Workload on page 373	Specify the workload object returned from the Get-DtWorkload cmdlet. See Get-DtWorkload on page 78.	true	false

Outputs

Guid on page 258

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FilesAndFolders

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The script then creates a workload on the server for a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. The connections for the server object are then closed.

Remove-DtJob

Deletes the job

Syntax

Remove-DtJob [-ServiceHost] <Server> [-JobId] <Guid> [[-DeleteOptions] <DeleteOptions>] [<CommonParameters>]

Remove-DtJob [-ServiceHost] <Server> [[-DeleteOptions] <DeleteOptions>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet deletes the specified job from the specified server. A running job will be stopped before it is deleted.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Delete Options	DeleteOptions on page 227	Specify the delete options available in DoubleTake.Jobs.Contract.DeleteOptions. Use the Windows PowerShell New-Object cmdlet to create this object.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}
```

```
$DtDeleteOptions = New-Object DoubleTake.Jobs.Contract.DeleteOptions

$DtDeleteOptions.DiscardTargetQueue = $true

Remove-DtJob -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -DeleteOptions
$DtDeleteOptions

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The delete options are stored in DtDeleteOptions, then the specific delete option DiscardTargetQueue is set to true. Finally the job is removed using the delete options. The connections for the server object are then closed.

Remove-DtPhysicalRule

Removes a physical rule

Syntax

Remove-DtPhysicalRule [-ServiceHost] <Server> [-WorkloadId] <Guid> [-Rule] <PhysicalRule> [<CommonParameters>]

Detailed Description

This cmdlet removes a physical rule from the specified workload on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your source server.	true	false
WorkloadId	Guid on page 258	Specify the workload GUID returned from the New-DtWorkload cmdlet using the workload type name parameter. See New-DtWorkload on page 99.	true	false
Rule	PhysicalRule on page 297	Use the Windows PowerShell New-Object cmdlet to create a physical rule object from DoubleTake.Common.Contract.PhysicalRule.	true	false

Outputs

ChangedItems on page 211

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FilesAndFolders

$DtPhysicalPath = New-Object DoubleTake.Common.Contract.PhysicalRule -Property @ {Path="C:\DirName"}

Remove-DtPhysicalRule -ServiceHost $DtServerObjectAlpha -WorkloadId $DtWorkloadGuid -Rule $DtPhysicalPath

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The script then creates a workload on the server for a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable

DtWorkloadGuid. A new object is created from Double-Take.Common.Contract.PhysicalRule to store the physical path C:\DirName in the variable DtPhysicalPath. Finally, the physical rule is removed from the workload on the server. The connections for the server object are then closed.

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FilesAndFolders

$DtWorkloadInfo=Get-DtWorkload -ServiceHost $DtServerObjectAlpha -WorkloadId $DtWorkloadGuid

$DtRemoveRule = $DtWorkloadInfo.PhysicalRules | Where-Object {$_.Path -eq "C:\DirNameToRemove"}

Remove-DtPhysicalRule -ServiceHost $DtServerObjectAlpha -WorkloadId $DtWorkloadGuid -Rule $DtRemoveRule

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The script then creates a workload on the server for a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. The workload information for the workload is then stored in DtWorkloadInfo. The physical rule within DtWorkloadInfo called C:\DirNameToRemove is then stored in DtRemoveRule. Finally, the physical rule DtRemoveRule is removed from the workload on the server. The connections for the server object are then closed.

Remove-DtSnapshot

Removes a snapshot

Syntax

Remove-DtSnapshot [-ServiceHost] <Server> [-JobId] <Guid> [-SnapshotId] <Guid> [-ConnectionId <Guid>] [<CommonParameters>]

Remove-DtSnapshot [-ServiceHost] <Server> [-JobId] <Guid> [-Snapshot] <SnapshotEntry> [-ConnectionId <Guid>] [<CommonParameters>]

Detailed Description

This cmdlet removes a Double-Take snapshot from the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
SnapshotId	Guid	Specify the snapshot GUID returned from the Get-DtSnapshot cmdlet. See Get-DtSnapshot on page 68.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
Snapshot	SnapshotEntry on page 333	Specify the snapshot entry object returned from the Get-DtSnapshot cmdlet. See Get-DtSnapshot on page 68.	true	false

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password
```

```
$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

$DtSnaps = Get-DtSnapshot -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

$FirstSnap=$DtSnaps | Select-Object -First 1

Remove-DtSnapshot -ServiceHost $DtServerObjectBeta -SnapshotId $FirstSnap.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The snapshots available for the job are stored in DtSnaps and then the first snapshot is stored in FirstSnap. That first snapshot is then deleted. The connections for the server object are then closed.

Repair-DtJobOptions

Fixes job option errors and warnings

Syntax

Repair-DtJobOptions [-ServiceHost] <Server> [-JobId] <Guid> [-JobOptions] <JobOptions> [-Step] <VerificationStep[]> [<CommonParameters>]

Repair-DtJobOptions [-ServiceHost] <Server> [-Source] <Server> [-JobType] <String> [-JobOptions] <JobOptions> [-Step] <VerificationStep[]> [-OtherServers <Server[]>] [<CommonParameters>]

Detailed Description

This cmdlet attempts to fix job option errors and warnings. For those errors and warnings that Double- Take cannot correct automatically, you will need to modify the job options manually, modify the source or target configuration, or perhaps select a different target. Use the first syntax for existing jobs and the second syntax for new jobs.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobOptions	JobOptions on page 268	Specify the JobOptions returned from the Get-DtRecommendedJobOptions or Get-DtUvraRecommendedJobOptions cmdlet. See Get-DtRecommendedJobOptions on page 58 or Get-DtUvraRecommendedJobOptions on page 72.	true	false
Step	VerificationStep on page 352	Specify the verification steps returned by the Get-DtVerificationStatus cmdlet. See Get-DtVerificationStatus on page 77.	true	true
Source	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92.	true	false
JobType	String	Specify the type of job from the following list. • **ClusterAwareDthv**—Cluster-aware agentless Hyper-V job • **ClusterAwareFilesAndFolders**—Cluster-	true	false

Name	Type	Description	Required	Pipeline Input
		aware files and folders job • **ClusterAwareHV2V**—Cluster-aware virtual to Hyper-V job • **ClusterAwareMultiSelectDthv**—Cluster-aware agentless Hyper-V job • **Diagnostics**—Throughput Diagnostic Utility job • **Dthv**—Agentless Hyper-V job • **Exchange**—Exchange job • **ExchangeClustered**—Cluster-aware Exchange job • **FilesAndFolders**—Files and folders job • **FullServerFailover**—Full server job • **Legacy**—Double-Take version 5.3 connections, Double-Take 6.0 connections made outside of the Double-Take Console or Double-Take PowerShell, or GeoCluster Replicated Disk Resource generated jobs • **MoveDataOnlyMigration**—Data migration job • **MoveServerMigration**—Full server migration job • **MultiSelectDthv**—Agentless Hyper-V job • **OrphanedConnection**—Orphaned job • **Sql**—SQL job • **SqlClustered**—Cluster-aware SQL job • **Uvra**—Full server to ESX appliance job • **V2V**—V to ESX or V to Hyper-V job • **Vra**—Full server to ESX or full server to Hyper-V job • **VraMove**—Full server to ESX migration or full server to Hyper-V migration job		
Other Servers	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. Specify multiple server objects in an array using the format @($server1, $server2).	false	false

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

$DtValidation = Confirm-DtJobOptions -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -
JobOptions $DtJob.Options

$DtStatus = Get-DtVerificationStatus -ServiceHost $DtServerObjectBeta -Token $DtValidation

$DtRepair = Repair-DtJobOptions -ServiceHost $DtTarget -JobId $DtJob.Id -JobOptions $DtJob.Options -
Step $DtStatus.Steps

Edit-DtJob -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -JobOptions $DtRepair.Options

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The job options used by the job are confirmed, and the validation result is stored in DtValidation. The details of the validation are stored in the variable DtStatus. Those items that can automatically be fixed are corrected. If the job options were modified in order to fix an issues, the updated job options are now contained in the variable $DtRepair. The updated job options are then applied to the job. The connections for the server object are then closed.

Request-DtOnlineActivation

Activates the license

Syntax

Request-DtOnlineActivation -Code <String> -ServerName <String> -ServerInformation <String> [-ServiceHost <Server>] [-EmailAddress <String>] [<CommonParameters>]

Detailed Description

This cmdlet activates the Double-Take license over the Internet using the server information returned by the Get-DtOnlineActivationRequest on page 46 cmdlet.

Parameters

Name	Type	Description	Required	Pipeline Input
Code	String	Specify the 24-character, alpha-numeric activation code (s) which applies the appropriate Double-Take license to your Double-Take server. Specify multiple codes in an array using the format @(code1, code2). You can also use the code that is returned by the Get-DtOnlineActivationRequest on page 46 cmdlet.	true	true
Server Name	String	Specify the name of the server or use the server name returned by the Get-DtOnlineActivationRequest on page 46 cmdlet.	true	true
Server Information	String	Specify the server information returned by the Get-DtOnlineActivationRequest on page 46 cmdlet.	true	true
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	false	false
Email Address	String	Specify a valid email address.	false	false

Outputs

ActivationInformation on page 190

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Get-DtOnlineActivationRequest -ServiceHost $DtServerObjectAlpha

Request-DtOnlineActivation -Code 1234-5678-9012-3456-7890-1234 -ServerName $DtServerObjectAlpha -
```

```
ServerInformation thnvkg9anmtjr5y27qgzhrmvqqbhyr5f7152pecq

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The server information for the online activation process is returned. The license is activated online. The connections for the server object are then closed.

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Get-DtOnlineActivationRequest -ServiceHost $DtServerObjectAlpha | Request-DtOnlineActivation | Set-DtActivationCode

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The server information for the online activation process is returned. The output from the Get-DtOnlineActivationRequest cmdlet is piped directly to Request-DtOnlineActivation, which activates the license online, and then that output (from Request-DtOnlineActivation) is piped directory to Set-DtActivationCode to set the activation key on the server.The connections for the server object are then closed.

Restart-DtReplicationService

Stop and restarts the Double-Take service

Syntax

Restart-DtReplicationService [-ServiceHost] <Server> [<CommonParameters>]

Detailed Description

This cmdlet stops and restarts the Double-Take service on the specified server. This cmdlet does not impact the Double-Take Management Service.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Restart-DtReplicationService -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the Double-Take service on the server is stopped and restarted. The connections for the server object are then closed.

Resume-DtJob

Resumes a paused job

Syntax

Resume-DtJob [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Resume-DtJob [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet resumes a paused job. All jobs from the same source to the same IP address on the target will be resumed.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Resume-DtJob -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable

DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The job is then resumed. The connections for the server object are then closed.

Resume-DtMirror

Resumes a paused mirror

Syntax

Resume-DtMirror [-ServiceHost] <Server> [-JobId] <Guid> [-ConnectionId <Guid>] [<CommonParameters>]

Resume-DtMirror [-ServiceHost] <Server> [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet resumes a paused mirror.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Resume-DtMirror -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The mirror for the job is then resumed. The connections for the server object are then closed.

Resume-DtTarget

Resumes Double-Take processing

Syntax

Resume-DtTarget [-ServiceHost] <Server> -All [<CommonParameters>]

Resume-DtTarget [-ServiceHost] <Server> [-JobId] <Guid> [-ConnectionId <Guid>] [<CommonParameters>]

Resume-DtTarget [-ServiceHost] <Server> -JobInfo <JobInfo> [-ConnectionId <Guid>] [<CommonParameters>]

Detailed Description

This cmdlet resumes Double-Take processing on the target.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
All	Switch Parameter	Execute the cmdlet on all jobs that are present	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password
```

```
Resume-DtTarget -ServiceHost $DtServerObjectBeta -All
Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. Double-Take processing on that server is then resumed. The connections for the server object are then closed.

Save-DtJobDiagnostics

Saves a diagnostics file

Syntax

Save-DtJobDiagnostics [-ServiceHost] <Server> [-JobId] <Guid[]> [<CommonParameters>]

Save-DtJobDiagnostics [-ServiceHost] <Server> -JobInfo <JobInfo[]> [<CommonParameters>]

Detailed Description

Saves a diagnostics file (also known as DTInfo) to the \Service\Data directory of your Double-Take installation on the specified server. This cmdlet is not applicable to Exchange jobs, SQL jobs, full server to ESX appliance jobs, or agentless Hyper-V jobs.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

None

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Save-DtJobDiagnostics -Servicehost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The diagnostics files is then saved on the server. The connections for the server object are then closed.

Set-DtActivationCode

Sets the Double-Take activation code

Syntax

Set-DtActivationCode [-ServiceHost] <Server> [-Code] <String[]> [-AdditionalCode <String[]>] [-ActivationKey <String>] [<CommonParameters>]

Detailed Description

This cmdlet sets the Double-Take activation code on the specified server. It also returns the Double-Take activation code validation information from the Get-DtActivationStatus cmdlet. See Get-DtActivationStatus on page 32.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	true
Code	String	Specify the 24-character, alpha-numeric activation code(s) which applies the appropriate Double-Take license to your Double-Take server. Specify multiple codes in an array using the format @(code1, code2).	true	true
Additional Code	String	Specify any additional activation codes, such as activation keys. Specify multiple codes in an array using the format @(code1, code2).	false	false
Activation Key	String	Specify the 24-character, alpha-numeric activation key which activates your Double-Take license.	false	false

Outputs

ActivationStatus on page 191

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Set-DtActivationCode -ServiceHost $DtServerObjectAlpha -Code 1234567890abcdefghij1234

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the activation code 1234567890abcdefghij1234 is applied to the server. The server returns the activation code information. The connections for the server object are then closed.

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Get-DtOnlineActivationRequest -ServiceHost $DtServerObjectAlpha | Request-DtOnlineActivation | Set-DtActivationCode

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The server information for the online activation process is returned. The output from the Get-DtOnlineActivationRequest cmdlet is piped directly to Request-DtOnlineActivation, which activates the license online, and then that output (from Request-DtOnlineActivation) is piped directory to Set-DtActivationCode to set the activation key on the server.The connections for the server object are then closed.

Set-DtBandwidthLimit

Sets bandwidth limiting

Syntax

Set-DtBandwidthLimit [-ServiceHost] <Server> [-JobId] <Guid> [-BandwidthLimit] <BandwidthLimit> [-ConnectionId <Guid>] [<CommonParameters>]

Set-DtBandwidthLimit [-ServiceHost] <Server> [-BandwidthLimit] <BandwidthLimit> [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet sets bandwidth limiting for the specified job .

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Bandwidth Limit	BandwidthLimit on page 203	Specify the bandwidth limit configuration from Get-BandwidthLimit. See Get-DtBandwidthLimit on page 33.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

None

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

$DtFixedBandwidth = Get-DtBandwidthLimit -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

$DtFixedBandwidth.Mode = "Fixed"

$DtFixedBandwidth.Limit = 100000

Set-DtBandwidthLimit -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -BandwidthLimit
$DtFixedBandwidth

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The bandwidth limiting configuration is then stored in DtFixedBandwidth. The Mode is then changed to fixed and the Limit is set to 100,000 bytes/second. Finally, the bandwidth settings are applied to the job. The connections for the server object are then closed.

Set-DtEmailNotificationOptions

Sets e-mail notification configuration

Syntax

Set-DtEmailNotificationOptions [-ServiceHost] <Server> [-Options] <EmailNotificationOptions> [<CommonParameters>]

Detailed Description

This cmdlet sets the Double-Take e-mail notification configuration for the specified server

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Options	EmailNotificationOptions on page 235	Specify the object returned from the Get-DtEmailNotificationOptions cmdlet. See Get-DtEmailNotificationOptions on page 37.	true	false

Outputs

EmailNotificationOptions on page 235

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtEmailOptions = Get-DtEmailNotificationOptions -ServiceHost $DtServerObject
$DtEmailOptions.Enabled = $true
$DtEmailOptions.SmtpServer = "mail.company.com"
Set-DtEmailNotificationOptions -ServiceHost $DtServerObjectAlpha -Options $DtEmailOptions
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the default Double-Take e-mail notification options are stored in the variable DtEmailOptions. Two of the options are then changed. The Enabled option is set to true which turns on the e-mail notification feature. The SMTP server is also configured for mail.company.com. Finally those changes for the email notification options are set on the server. The connections for the server object are then closed.

Set-DtJobCredentials

Updates credentials

Syntax

Set-DtJobCredentials [-ServiceHost] <Server> [-JobId] <Guid> [-Source <PSCredential>] [-Target <PSCredential>] [-OtherServers <Server[]>] [<CommonParameters>]

Set-DtJobCredentials [-ServiceHost] <Server> [-Source <PSCredential>] [-Target <PSCredential>] [-OtherServers <Server[]>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

Updates the credentials for the source, appliance, and VMware host servers used in the specified job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer or New-DtUvraServer cmdlet. See New-DtServer on page 92 or New-DtUvraServer on page 97.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Source	PSCredential on page 302	Specify the credential object returned from the Windows PowerShell Get-Credential cmdlet. This password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.	false	false
Target	PSCredential on page 302	Specify the credential object returned from the Windows PowerShell Get-Credential cmdlet. This password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.	fasle	false
Other Servers	Server on page 322	Specify the server object returned from the New-DtServer or New-DtUvraServer cmdlet. See New-DtServer on page 92 or New-DtUvraServer on page 97. Specify the server object in an array using the format @	false	false

Name	Type	Description	Required	Pipeline Input
		($server).		
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

None

Examples

```
$DtCredentials = Get-Credential domain\administrator

$DtServerObjectAlpha = New-DtServer -Name alpha -Credential $DtCredentials

$DtServerObjectBeta = New-DtServer -Name beta -Credential $DtCredentials

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Set-DtJobCredentials -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -Source $DtCredentials -
Target $DtCredentials

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

You will be prompted for credentials for the domain\administrator account and they will be stored in DtCredentials. Then a server object is created for the servers alpha and beta using the stored credentials. The objects are stored in DtServerObjectAlpha and DtServerObjectBeta, respectively. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Finally, the credentials are updated for the job using the stored credentials. The connections for the server object are then closed.

Set-DtLogicalItemSelection

Adds or removes logical items

Syntax

Set-DtLogicalItemSelection [-ServiceHost] <Server> [-WorkloadId] <Guid> [-LogicalPath] <String> [-Unselect] [<CommonParameters>]

Detailed Description

This cmdlet adds or removes a logical items for the specified workload for the specified server. Adding or removing logical items will add or remove physical rules depending on the workload type.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your source server.	true	false
WorkloadId	Guid on page 258	Specify the workload GUID returned from the New-DtWorkload cmdlet using the workload type name parameter. See New-DtWorkload on page 99.	true	false
Logical Path	String	Specify the path of an item returned from Get-DtLogicalItem. See Get-DtLogicalItem on page 43.	true	false
Unselect	Switch Parameter	Specify this option if you want to remove the logical item.	false	false

Outputs

ChangedItems on page 211

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtWorkloadGuid = New-DtWorkload -ServiceHost $DtServerObjectAlpha-WorkloadTypeName FilesAndFolders

$DtLogicalItems = Get-DtLogicalItem -ServiceHost $DtServerObjectAlpha -WorkloadId $DtWorkloadGUID

Set-DtLogicalItemSelection -ServiceHost $DtServerObjectAlpha -WorkloadId $DtWorkloadGuid -LogicalPath $DtLogicalItem[0].Path

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. The script then creates a workload on the server for

a files and folders job, returning a global unique ID for the workload, and assigns that ID to the variable DtWorkloadGuid. The logical items associated with the workload type and the server are then stored in the variable DtLogicalItems. Finally, the first logical item in DtLogicalItems is added to the workload. The connections for the server object are then closed.

Set-DtOption

Sets server or job options

Syntax

Set-DtOption [-ServiceHost] <Server> [-Setting] <Hashtable>] [<CommonParameters>]

Set-DtOption [-ServiceHost] <Server> [-Name] <String> -IntValue <Int64> [<CommonParameters>]

Set-DtOption [-ServiceHost] <Server> [-Name] <String> -MultiStringValue <String[]> [<CommonParameters>]

Set-DtOption [-ServiceHost] <Server> [-Name] <String> -StringValue <String> [<CommonParameters>]

Detailed Description

This cmdlet sets the value of the Double-Take server or job option for the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Setting	Hashtable	Specify a hash table using the format @{option1=value1; option2=value2}. See Server and job settings on page 421 for details on each job and server option.	true	false
Name	String	Specify the name of the job or server option. See Server and job settings on page 421 for details on each job and server option.	true	false
IntValue	Int64	Specify an integer value for the server or job option. See Server and job settings on page 421 for details on each job and server option.	true	false
MultiString Value	String	Specify multiple string (text) values for the server or job option in an array using the format @(string1, string2).	true	false
StringValue	String	Specify a single string (text) value for the server or job option	true	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Set-DtOption -ServiceHost $DtServerObjectAlpha -Setting @{MaxChecksumBlocks=64;
MirrorChunkSize=131072}

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the server settings MaxChecksumBlocks and MirrorChunkSize are set to 64 and 131072, respectively, on the server. The connections for the server object are then closed.

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Set-DtOption -Servicehost $DtServerObjectAlpha -Name MirrorChunkSize -IntValue 64

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then it sets the MirrorChunkSize server setting to 64. The connections for the server object are then closed.

See Viewing and setting job and server options on page 409 for a sample script that gathers and sets several Double-Take job and server options.

Set-DtPathBlocking

Blocks writing on the target

Syntax

Set-DtPathBlocking [-ServiceHost] <Server> [-SourceAddress] <String> [[-Mode] <PathBlockingMode>] [<CommonParameters>]

Detailed Description

This cmdlet blocks writing to the replica source data located on the target, keeping the data from being changed outside of Double-Take processing.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
Source Address	String	Specify the IP address of the source, including the port, for example, 123.123.123.123:6320.	true	false
Mode	Path Blocking Mode	Specify Blocked or Unblocked.	false	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Set-DtPathBlocking -ServiceHost $DtServerObjectAlpha -SourceAddress "112.42.74.29:6320" -Mode Blocked
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then it sets the target paths associated with the replica data from the specified source IP address to blocked. The connections for the server object are then closed.

Set-DtScriptCredentials

Sets credentials

Syntax

Set-DtScriptCredentials [-ServiceHost] <Server> [-Credential] <PSCredential> [<CommonParameters>]

Set-DtScriptCredentials [-ServiceHost] <Server> [-UserName] <String> [-Password] <String> [[-Domain] <String>]

Detailed Description

This cmdlet sets the credentials for Double-Take to use when running scripts on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Credential	PSCredential on page 302	Specify the credential object returned from the Windows PowerShell Get-Credential cmdlet. This password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.	true	false
UserName	String	Specify a user name.	true	false
Password	String	Specify the password associated with the user you have entered. This password will be visible in plain text.	true	false
Domain	String	Specify the domain name.	false	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtCredentials = Get-Credential domain\administrator

Set-DtScriptCredentials -ServiceHost $DtServerObjectAlpha -Credential $DtCredentials
```

```
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then you will be prompted for credentials for the domain\administrator account and those credentials will be stored in the variable DtCredentials. Finally, the credentials used for Double-Take scripts on the server will be set to the stored credentials. The connections for the server object are then closed.

Set-DtServerCredential

Changes server credentials

Syntax

Set-DtServerCredential -Input <Server> -Credential <PSCredential> [<CommonParameters>]

Detailed Description

This cmdlet changes the credentials associated with the server object that has already been created using the New-DtServer cmdlet. See New-DtServer on page 92 for more details on creating a server object.

Parameters

Name	Type	Description	Required	Pipeline Input
Input	String or Server on page 322	Specify the name or IP address of the server, cluster, or cluster node. Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92.	true	false
Credential	PSCredential on page 302	Specify the credential object returned from the Windows PowerShell Get-Credential cmdlet. This password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.	true	false

Outputs

Server on page 322

Examples

```
$DtCredentialEncrypted = Get-Credential
Set-DtServerCredential -Input alpha -Credential $DtCredential
```

User credentials are stored in a variable called $DtCredential. The script will prompt you to supply the username and password and the credentials will be encrypted. Then the stored credentials are used to update the current credentials on the server alpha.

Start-DtJob

Starts a job

Syntax

Start-DtJob [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Start-DtJob [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet starts the specified job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Start-DtJob -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable

DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The job is then started. The connections for the server object are then closed.

Start-DtJobFailback

Starts failback

Syntax

Start-DtJobFailback [-ServiceHost] <Server> [-JobId] <Guid> [-FailbackOptions] <FailbackOptions> [<CommonParameters>]

Start-DtJobFailback [-ServiceHost] <Server> [-FailbackOptions] <FailbackOptions> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet starts failback for the specified job using the specified failback options.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Failback Options	FailbackOptions on page 241	Specify the failback options returned from the Get-DtRecommendedFailbackOptions cmdlet. See Get-DtRecommendedFailbackOptions on page 54.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}
```

```
$DtFailbackOptions = Get-DtRecommendedFailbackOptions -ServiceHost $DtServerObjectBeta -JobId
$DtJobForAlpha.Id

Start-DtJobFailback -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -FailbackOptions
$DtFailbackOptions

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The failback options are stored in DtFailbackOptions. Failback is then started using the failback options. The connections for the server object are then closed.

Start-DtJobFailover

Starts failover

Syntax

Start-DtJobFailover [-ServiceHost] <Server> [-JobId] <Guid> [-FailoverOptions] <FailoverOptions> [<CommonParameters>]

Start-DtJobFailover [-ServiceHost] <Server> [-FailoverOptions] <FailoverOptions> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet starts failover for the specified job using the specified failover options.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Failover Options	FailoverOptions on page 246	Specify the failover options returned from the Get-DtRecommendedFailoverOptions cmdlet. See Get-DtRecommendedFailoverOptions on page 56.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}
```

```
$DtFailoverOptions = Get-DtRecommendedFailoverOptions -ServiceHost $DtServerObjectBeta -JobId
$DtJobForAlpha.Id

$DtFailoverOptions.FailoverOptions.PerformTestFailover = $true

Start-DtJobFailover -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -FailoverOptions
$DtFailoverOptions.FailoverOptions

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The failover options are stored in DtFailoverOptions, and then the PerformTestFailover option is set to true. Failover is then started using the failover options. The connections for the server object are then closed.

Start-DtJobRestore

Starts restoration

Syntax

Start-DtJobRestore [-ServiceHost] <Server> [-JobId] <Guid> [-RestoreOptions] <RestoreOptions> [<CommonParameters>]

Start-DtJobRestore [-ServiceHost] <Server> [-RestoreOptions] <RestoreOptions> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet starts the restoration process for the specified job using the specified restoration options.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Restore Options	RestoreOptions on page 312	Specify the restoration options returned from the Get-DtRecommendedRestoreOptions cmdlet. See Get-DtRecommendedRestoreOptions on page 62.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
```

```
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

$DtRestoreOptions = Get-DtRecommendedRestoreOptions -ServiceHost $DtServerObjectBeta -JobId
$DtJobForAlpha.Id -RestoreTarget $DtServerObjectAlpha

$DtRestoreOptions.RestoreOptions.RestoreParameters.ProcessOrphans = $true

Start-DtJobRestore -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -RestoreOptions
$DtRestoreOptions

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The restoration options are stored in DtRestoreOptions, and then the ProcessOrphans option is set to true. Restoration is then started using the restoration options. The connections for the server object are then closed.

Start-DtJobReverse

Starts reverse

Syntax

Start-DtJobReverse [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Start-DtJobReverse [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet starts the reverse process for the specified job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Start-DtJobReverse -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable

DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The job is then reversed. The connections for the server object are then closed.

Start-DtMirror

Starts mirroring

Syntax

Start-DtMirror [-ServiceHost] <Server> [-JobId] <Guid> [-MirrorParameters] <MirrorParameters> [-ConnectionId <Guid>] [<CommonParameters>]

Start-DtMirror [-ServiceHost] <Server> [-MirrorParameters] <MirrorParameters> [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet starts mirroring on the specified job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Mirror Parameters	MirrorParameters on page 280	Specify the mirror options available in DoubleTake.Core.Contract.Connection.MirrorParameters. Use the Windows PowerShell New-Object cmdlet to create this object.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

$DtMirrorChecksum = New-Object DoubleTake.Core.Contract.Connection.MirrorParameters

$DtMirrorChecksum.ComparisonCriteria = "Checksum"

Start-DtMirror -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -MirrorParameters
$DtMirrorChecksum

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The mirror options are stored in DtMirrorChecksum and then the ComparisonCriteria value is changed to checksum. Finally the mirror is started for the job using the mirroring options. The connections for the server object are then closed.

Start-DtOrphansProcessing

Starts orphan processing

Syntax

Start-DtOrphansProcessing [-ServiceHost] <Server> [-JobId] <Guid> [-ConnectionId <Guid>] [<CommonParameters>]

Start-DtOrphansProcessing [-ServiceHost] <Server> [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet starts orphan files processing on the specified job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}
```

```
Start-DtOrphansProcessing -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id
Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Orphan file processing is then started for the job. The connections for the server object are then closed.

Start-DtReplication

Starts replication

Syntax

Start-DtReplication [-ServiceHost] <Server> [-JobId] <Guid> [-ConnectionId <Guid>] [<CommonParameters>]

Start-DtReplication [-ServiceHost] <Server> [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet starts replication on the specified job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
ConnectionId	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Start-DtReplication -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id
```

```
Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Replication is then started for the job. The connections for the server object are then closed.

Start-DtVerify

Starts verification

Syntax

Start-DtVerify [-ServiceHost] <Server> [-JobId] <Guid> [-Synchronize] [-Newer] [-Checksum] [-ProcessOrphans] [-ConnectionId <Guid>] [<CommonParameters>]

Start-DtVerify [-ServiceHost] <Server> [-Synchronize] [-Newer] [-Checksum] [-ProcessOrphans] [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet starts the Double-Take verification process to check that the replica source data on the target is identical to the actual data on the source

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Synchronize	Switch Parameter	Mirrors to the target any protected files that are different on the source. Without this option, the verification process will only verify the data and generate a verification log file, but it does not remirror any files that are different on the source and target.	false	false
Newer	Switch Parameter	If you are mirroring files to the target during the verification process with the synchronize option, this option will only mirror files that are newer on the source than on the target. If you are using a database application, do not use this option unless you know for certain that you need it. With database applications, it is critical that all files, not just some of the file that might be newer, get mirrored.	false	false
Checksum	Switch Parameter	If you are mirroring files to the target during the verification process with the synchronize option, this option will have the verification process perform a block checksum comparison to determine which	false	false

Name	Type	Description	Required	Pipeline Input
		blocks are different.		
Process Oprhans	Switch Parameter	If you are mirroring files to the target during the verification process with the synchronize option, this option will delete orphaned files on the target.	false	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

None

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Start-DtVerify -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -Synchronize -Newer -Checksum
-ProcessOrphans

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Verification is then started for the job. The connections for the server object are then closed.

Stop-DtJob

Stops a job

Syntax

Stop-DtJob [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Stop-DtJob [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet stops the specified job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Stop-DtJob -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable

DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The job is then stopped. The connections for the server object are then closed.

Stop-DtMirror

Stops mirroring

Syntax

Stop-DtMirror [-ServiceHost] <Server> [-JobId] <Guid> [-ConnectionId <Guid>] [<CommonParameters>]

Stop-DtMirror [-ServiceHost] <Server> [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet stops mirroring on the specified job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Stop-DtMirror -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The mirror is then stopped for the job. The connections for the server object are then closed.

Stop-DtReplication

Stops replication

Syntax

Stop-DtReplication [-ServiceHost] <Server> [-JobId] <Guid> [-ConnectionId <Guid>] [<CommonParameters>]

Stop-DtReplication [-ServiceHost] <Server> [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet stops replication on the specified job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Stop-DtReplication -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id
```

```
Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Replication is then stopped for the job. The connections for the server object are then closed.

Stop-DtReplicationService

Stops the Double-Take service

Syntax

Stop-DtReplicationService [-ServiceHost] <Server> [<CommonParameters>]

Detailed Description

This cmdlet stops the Double-Take service on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Stop-DtReplicationService -ServiceHost $DtServerObjectAlpha

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the Double-Take service is stopped on the server. The connections for the server object are then closed.

Suspend-DtJob

Pauses a job

Syntax

Suspend-DtJob [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Suspend-DtJob [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet paused a job. All jobs from the same source to the same IP address on the target will be paused.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Suspend-DtJob -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable

DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The job is then suspended. The connections for the server object are then closed.

Suspend-DtMirror

Pauses mirroring

Syntax

Suspend-DtMirror [-ServiceHost] <Server> [-JobId] <Guid> [-ConnectionId <Guid>] [<CommonParameters>]

Suspend-DtMirror [-ServiceHost] <Server> [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet pauses mirroring.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Suspend-DtMirror -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The mirror for the job is then paused. The connections for the server object are then closed.

Suspend-DtTarget

Pauses Double-Take processing

Syntax

Suspend-DtTarget [-ServiceHost] <Server> [-JobId] <Guid> [-ConnectionId <Guid>] [<CommonParameters>]

Suspend-DtTarget [-ServiceHost] <Server> -All [<CommonParameters>]

Suspend-DtTarget [-ServiceHost] <Server> -JobInfo <JobInfo> [-ConnectionId <Guid>] [<CommonParameters>]

Detailed Description

This cmdlet pauses Double-Take processing on the target. Incoming Double-Take data from the source will be queued on the target. All active jobs to the target will complete the operations already in progress. Any new operations will be queued on the target until the target is resumed. The data will not be committed until the target is resumed. Pausing the target only pauses Double-Take processing, not the entire server.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
All	Switch Parameter	Execute the cmdlet on all jobs that are present	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Suspend-DtTarget -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Double-Take processing for the job is then paused. The connections for the server object are then closed.

Test-DtActiveDirectoryCredentials

Tests credentials against Active Directory

Syntax

Test-DtActiveDirectoryCredentials [-ServiceHost] <Server> [-Credential] <PSCredential> [<CommonParameters>]

Test-DtActiveDirectoryCredentials [-ServiceHost] <Server> [-UserName] <String> [-Password] <String> [[-Domain] <String>]

Detailed Description

This cmdlet tests if the specified credentials have privileges to update Actiev Directory on the specified server's domain.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Credential	PSCredential on page 302	Specify the credential object returned from the Windows PowerShell Get-Credential cmdlet. This password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.	false	false
UserName	String	Specify a user name.	true	false
Password	String	Specify the password associated with the user you have entered. This password will be visible in plain text.	true	false
Domain	String	Specify the domain name.	false	false

Outputs

Boolean

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtCredentials = Get-Credential domain\administrator
```

```
Test-DtActiveDirectoryCredentials -ServiceHost $DtServerObjectAlpha -Credential $DtCredentials
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then you will be prompted for credentials for the domain\administrator account and those credentials will be stored in the variable DtCredentials. Finally, the stored credentials will be tested to see if they can update Active Directory on the server's domain. The connections for the server object are then closed.

Test-DtEmailNotification

Tests e-mail configuration

Syntax

Test-DtEmailNotification [-ServiceHost] <Server> [-Options] <EmailNotificationOptions> [-To] <String> [-Body] <String> [<CommonParameters>]

Detailed Description

Tests the e-mail options configured with Set-DtEmailNotificationOptions by attempting to send an e-mail to the specified recipient. See Set-DtEmailNotificationOptions on page 124

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Options	EmailNotificationOptions on page 235	Specify the object returned from the Get-DtEmailNotificationOptions cmdlet. See Get-DtEmailNotificationOptions on page 37.	true	false
To	String	Specify the e-mail address that the test Double-Take e-mail message should be sent to. Multiple addresses can be separated by a comma.	true	false
Body	String	Specify the text of the test Double-Take email message.	true	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtEmailOptions = Get-DtEmailNotificationOptions -ServiceHost $DtServerObject
$DtEmailOptions.Enabled = $true
$DtEmailOptions.SmtpServer = "mail.company.com"
Set-DtEmailNotificationOptions -ServiceHost $DtServerObjectAlpha -Options $DtEmailOptions
```

```
Test-DtEmailNotification -ServiceHost $DtServerObjectAlpha -Options $DtEmailOptions -To
"administrator@mail.company.com" -Body "This is a test Double-Take message."

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the default Double-Take e-mail notification options are stored in the variable DtEmailOptions. Two of the options are then changed. The Enabled option is set to true which turns on the e-mail notification feature. The SMTP server is also configured for mail.company.com. Those changes for the email notification options are set on the server. Finally a test message is sent to the administrator@mail.company.com addresses with the specified message text using the configured e-mail notification options. The connections for the server object are then closed.

Test-DtScript

Tests the specified script

Syntax

Test-DtScript [-ServiceHost] <Server> [-Path] <String> [[-Arguments] <String>] [[-InteractionMode] <DesktopInteractionMode>] [<CommonParameters>]

Detailed Description

This cmdlet tests the specified script on the specified server using the credentials from Set-DtScriptCredentials on page 132. If necessary, manually undo any changes that you do not want after testing the script.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Path	String	Specify the full path and script name	true	false
Arguments	String	Specify any arguments that need to be passed to the script.	false	false
Interaction Mode	DesktopInteractionMode on page 228	Specify if the script processing will be displayed on the screen, by using the value Interact, or if the script will execute silently in the background, by using the value None.	false	false

Outputs

Int32

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Test-DtScript -ServiceHost $DtServerObjectAlpha -Path "C:\PathDir\ScriptName" -Arguments "arg1 arg2" -InteractionMode Interact

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then the script called ScriptName (located in C:\PathDir) is run, using the arguments arg1 and arg2. The script will display on screen. The connections for the server object are then closed.

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

Test-DtScript -ServiceHost $DtServerObjectAlpha -Path
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -Arguments "-File ""C:\my
scripts\myscript.ps1"" ""-Arg1 arg1 -Arg2 arg2"" -ExecutionPolicy RemoteSigned" -InteractionMode None

Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then PowerShell is launched and the arguments passed to PowerShell are the PowerShell script myscript.ps1 (located in C:\my scripts) and the arguments ars1 and arg2. The PowerShell execution policy is set to RemoteSigned so the PowerShell script will execute. The script is set to run silently in the background. The connections for the server object are then closed.

Test-DtScriptCredentials

Tests credentials

Syntax

Test-DtScriptCredentials [-ServiceHost] <Server> [-Credential <PSCredential>] [<CommonParameters>]

Test-DtScriptCredentials [-ServiceHost] <Server> [-UserName] <String> [-Password] <String> [[-Domain] <String>]

Detailed Description

This cmdlet tests the specified credentials on the specified server to confirm if they have administrative rights

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost could be your source or target server.	true	false
Credential	PSCredential on page 302	Specify the credential object returned from the Windows PowerShell Get-Credential cmdlet. This password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.	false	false
UserName	String	Specify a user name.	true	false
Password	String	Specify the password associated with the user you have entered. This password will be visible in plain text.	true	false
Domain	String	Specify the domain name.	false	false

Outputs

Boolean

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password

$DtCredentials = Get-Credential domain\administrator

Test-DtScriptCredentials -ServiceHost $DtServerObjectAlpha -Credential $DtCredentials
```

```
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then you will be prompted for credentials for the domain\administrator account and those credentials will be stored in the variable DtCredentials. Finally, the credentials will be tested to confirm if they have administrative rights on the server. The connections for the server object are then closed.

Undo-DtJobFailover

Starts undo failover

Syntax

Undo-DtJobFailover [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Undo-DtJobFailover [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet starts the undo failover process for the specified job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Undo-DtJobFailover -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable

DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. The undo failover process is then started. The connections for the server object are then closed.

Uninstall-DoubleTake

Uninstalls Double-Take

Syntax

Uninstall-DoubleTake [-RemoteServer] <Server> [-AsJob] [<CommonParameters>]

Detailed Description

This cmdlet uninstalls Double-Take on the specified server.

Parameters

Name	Type	Description	Required	Pipeline Input
Remote Server	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92.	true	false
AsJob	Switch Parameter	Specify if you want the uninstallation to occur asynchronously in the background, returning the PowerShell command immediately. You can get the status of each uninstallation using the Windows PowerShell Get-Job command. Without this parameter, each uninstallation specified will be executed synchronously and the current activity of the current uninstallation will be displayed.	false	false

Outputs

None

Examples

```
$DtServerObjectAlpha= New-DtServer -Name alpha -UserName domain\administrator -Password password
Uninstall-Doubletake -RemoteServer $DtServerObjectAlpha
Disconnect-DtServer -ServiceHost $DtServerObjectAlpha
```

A server object is created for the server alpha using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectAlpha. Then Double-Take is uninstalled from the server. The connections for the server object are then closed.

Update-DtShares

Updates shares

Syntax

Update-DtShares [-ServiceHost] <Server> [-JobId] <Guid> [<CommonParameters>]

Update-DtShares [-ServiceHost] <Server> -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet updates source share information on the target for the specified job.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

ActivityToken on page 195

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password

$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Update-DtShares -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable

DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Shares are then updated on the target for the job. The connections for the server object are then closed.

Wait-DtMirrorComplete

Waits for the mirroring process to complete

Syntax

Wait-DtMirrorComplete [-ServiceHost] <Server> [-JobId] <Guid> [-PollingInterval <Int32>] [-ConnectionId <Guid>] [<CommonParameters>]

Wait-DtMirrorComplete [-ServiceHost] <Server> [-PollingInterval <Int32>] [-ConnectionId <Guid>] -JobInfo <JobInfo> [<CommonParameters>]

Detailed Description

This cmdlet waits for the mirroring process to complete before processing any additional cmdlets.

Parameters

Name	Type	Description	Required	Pipeline Input
Service Host	Server on page 322	Specify the server object returned from the New-DtServer cmdlet. See New-DtServer on page 92. For this cmdlet, the -ServiceHost should be your target server.	true	false
JobId	Guid on page 258	Specify the job GUID returned from the New-DtJob cmdlet or the Id within the job information returned from the Get-DtJob cmdlet. See New-DtJob on page 89 and Get-DtJob on page 39.	true	false
Polling Interval	Int32	Specify the amount of time, in hh:mm:ss, to wait before checking to see if the mirror has completed.	false	false
Connection Id	ConnectionId on page 216	Specify the connection ID returned from the Get-DtConnectionIds cmdlet. See Get-DtConnectionIds on page 35.	false	false
JobInfo	JobInfo on page 266	Specify the job information returned from the Get-DtJob cmdlet. The job information can be piped from the Get-DtJob cmdlet and used in this cmdlet. See Get-DtJob on page 39.	true	true

Outputs

MirrorState on page 281

Examples

```
$DtServerObjectBeta = New-DtServer -Name beta -UserName domain\administrator -Password password
```

```
$DtJobForAlpha = Get-DtJob -ServiceHost $DtServerObjectBeta | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}

Wait-DtMirrorComplete -ServiceHost $DtServerObjectBeta -JobId $DtJobForAlpha.Id -PollingInterval
"00:05:00"

Disconnect-DtServer -ServiceHost $DtServerObjectBeta
```

A server object is created for the server beta using the domain\administrator and password credentials. It assigns the server object to the variable called DtServerObjectBeta. The job(s) are retrieved from DtServerObjectBeta, but only the job information where the source machine name is equivalent to the name stored in the variable DtServerObjectAlpha is retrieved. That information is then stored in the variable DtJobForAlpha. Finally, the script will wait for mirroring to complete before continuing. The script will check ever five minutes to see if mirroring is complete. The connections for the server object are then closed.

Chapter 3 Classes and enumerations

The following classes and enumerations are used in the Double-Take PowerShell cmdlets.

AccessLevel

Returned by

Get-DtAccessLevel on page 31

Parameter of

CoreConnectionDetails on page 218

Properties

Name	Enumeration
Unknown	-1
NoAccess	0
MonitorOnlyAccess	1
FullAccess	2

ActionStatus

Parameter of

JobAction on page 265

Properties

Name	Enumeration
Pending	0
Running	1
Completed	2
Cancelled	3
Faulted	4

ActivationCode

Parameter of

ActivationStatus on page 191

Properties

Name	Type
Attributes	Attributes [ ] on page 200
Code	String
ExpirationDate	DateTime
IsEvaluation	Boolean
IsExpired	Boolean
IsNodeLocked	Boolean
IsValid	Boolean
LicenseType	LicenseType on page 272
MajorVersion	Int32
MinorVersion	Int32
ProductCode	Int32
ProductName	String
SerialNumber	Int32

ActivationCodeInfo

Parameter of

V2VQualificationResults on page 349

Properties

Name	Type
ActivationCode	String
DaysToExpire	Int32
IsEval	Boolean
IsNodeLocked	Boolean
IsValid	Boolean

ActivationInformation

Returned by

Request-DtOnlineActivation on page 109

Properties

Name	Type
ActivationKey	String
Code	String
Error	String
Quantity	Int32
Server	String
ServerInformation	String

ActivationStatus

Returned by

Get-DtActivationStatus on page 32, Set-DtActivationCode on page 120

Properties

Name	Type
AddOnCodes	ActivationCode [ ] on page 188
Codes	ActivationCode [ ] on page 188
IsNodeLocked	Boolean
IsValid	Boolean

ActiveDirectoryOptions

Parameter of

MonitorConfiguration on page 282

Properties

Name	Enumeration
None	0
FailoverHostName	1
FailbackHostName	2

ActivityCompletionStatus

Parameter of

ActivityStatusEntry on page 194, VerificationStep on page 352

Properties

Name	Enumeration
InProgress	0
Success	1
Warning	2
Error	3

ActivityStatusEntry

Returned by

Get-DtJobActionStatus on page 41

Parameter of

RepairStatus on page 308, VerificationStatus on page 351

Properties

Name	Type
Token	ActivityToken on page 195
TimeStamp	DateTimeOffset
RequesterUserName	String
MessageId	String
MessageFormatParameters	String
Duration	TimeSpan
Status	ActivityCompletionStatus on page 193

ActivityToken

Returned by

Confirm-DtJobOptions on page 25, Edit-DtJob on page 29, Invoke-DtQueueTask on page 84, Remove-DtJob on page 100, Remove-DtSnapshot on page 104, Repair-DtJobOptions on page 106, Resume-DtJob on page 112, Resume-DtMirror on page 114, Resume-DtTarget on page 116, Start-DtJob on page 135, Start-DtJobFailback on page 137, Start-DtJobFailover on page 139, Start-DtJobRestore on page 141, Start-DtJobReverse on page 143, Start-DtMirror on page 145, Start-DtOrphansProcessing on page 147, Start-DtReplication on page 149, Start-DtVerify on page 151, Stop-DtJob on page 153, Stop-DtMirror on page 155, Stop-DtReplication on page 157, Suspend-DtJob on page 160, Suspend-DtMirror on page 162, Suspend-DtTarget on page 164, Undo-DtJobFailover on page 174, Update-DtShares on page 177

Parameter of

ActivityStatusEntry on page 194, Get-DtVerificationStatus on page 77

Properties

Name	Type
Id	Guid
ActivityNameId	String
ActivityNameFormatParameters	String

ApplicationOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
DagTargetPublicFolderDataPath	String
DnsRecordLocks	DnsRecordLock on page 231
ExchangeCredentials	Credentials on page 224
IsDag	Boolean
MonitoredServiceRepeatCount	Int32
MonitoredServices	ServiceInformation on page 327
MonitorScript	String
RestartService	Boolean
SourceDomain	String
SourceName	String
TargetDomain	String
TargetName	String
TestPostFailoverScript	String
TestPreFailbackScript	String

ArchiveCriteria

Parameter of

ArchiveSchedule on page 199

Properties

Name	Type
Age	TimeSpan
DiskUtilizationPercentage	Int32
LogErrors	Boolean
MinimumFileSize	Int64
Paths	String

ArchiveOption

Parameter of

RestoreParameters on page 313

Properties

Name	Enumeration
None	0
ArchiveBeforeRestore	1
RecallBeforeRestore	2

ArchiveSchedule

Parameter of

Schedule on page 318

Properties

Name	Type
Critera	ArchiveCriteria on page 197
Days	Weekdays on page 372
IsEnabled	Boolean
StartTime	DateTime

Attributes

Parameter of

ActivationCode on page 188

Properties

Name	Type
Name	String
Value	Int64

BandwidthEntry

Parameter of

BandwidthOptions on page 204

Properties

Name	Type
DaysOfWeek	Weekdays on page 372
EndTime	DateTime
EntryType	BandwidthEntryType on page 202
IsUnlimited	Boolean
Limit	Int64
Name	String
StartTime	DateTime

BandwidthEntryType

Parameter of

BandwidthEntry on page 201

Properties

Name	Enumeration
Daytime	0
Overnight	1

BandwidthLimit

Returned by

Get-DtBandwidthLimit on page 33

Parameter of

Set-DtBandwidthLimit on page 122

Properties

Name	Type
Limit	Int64
Mode	BandwidthScheduleMode on page 207

BandwidthOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
Entries	BandwidthEntry [ ] on page 201
Limit	Int64
Mode	BandwidthScheduleMode on page 207
Specifications	BandwidthSpecification [ ] on page 208

BandwidthSchedule

Parameter of

Schedule on page 318

Properties

Name	Type
Current	BandwidthScheduleEntry [ ] on page 206
Entries	BandwidthScheduleEntry [ ] on page 206
Mode	BandwidthScheduleMode on page 207
Specifications	BandwidthSpecification on page 208

BandwidthScheduleEntry

Parameter of

BandwidthSchedule on page 205

Properties

Name	Type
DaysOfWeek	Weekdays on page 372
IsUnlimited	Boolean
Limit	Int64
Name	String
StartTime	DateTime

BandwidthScheduleMode

Parameter of

BandwidthLimit on page 203, BandwidthOptions on page 204, BandwidthSchedule on page 205

Properties

Name	Enumeration
NotLimited	0
Fixed	1
Scheduled	2

BandwidthSpecification

Parameter of

BandwidthOptions on page 204, BandwidthSchedule on page 205

Properties

Name	Type
Key	String
Type	BandwidthSpecificationType on page 209
Value	Int64

BandwidthSpecificationType

Parameter of

BandwidthSpecification on page 208

Properties

Name	Enumeration
LAN	0
WAN	1

BlockingMode

Parameter of

PathBlocking on page 294

Properties

Name	Enumeration
Blocked	0
Unblocked	1

ChangedItems

Returned by

Add-DtPhysicalRule on page 18, Remove-DtPhysicalRule on page 102, Set-DtLogicalItemSelection on page 127

Parameter of

Add-DtPhysicalRule on page 18, Add-DtUvraPhysicalRule on page 20, Remove-DtPhysicalRule on page 102

Properties

Name	Type
LogicalItems	LogicalItems [ ] on page 273
LogicalRules	String [ ]
PhysicalItems	PhysicalItem [ ] on page 296
PhysicalRules	PhysicalRule [ ] on page 297

ClusterFilesAndFoldersQualifcationResults

Parameter of

JobQualificationResults on page 269

Properties

Name	Type
TargetFileServerQualificationResults	TargetFileServerQualificationResults on page 339

ClusterOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
DependOnResources	String
OriginalSourceClusterId	String
SourceGroupName	String
SourceIsCluster	Boolean
TargetClusterResourceNamePrefix	String
TargetClusterStoragePath	String
TargetGroupName	String
TargetIsCluster	Boolean
TargetVirtualServers	String

ClusterResourceState

Parameter of

CoreConnectionDetails on page 218

Properties

Name	Enumeration
Uninitialized	0
OfflinePending	1
Offline	2
OnlinePending	3
Online	4
ResourceNotFound	5
Reconnected	6
Connected	7

CompressionLevel

Parameter of

ConnectionStartParameters on page 217, CoreQualifcationResults on page 223, RestoreParameters on page 313

Properties

Name	Type
Algorithm	Int32
Level	Int32

Notes

The algorithm and level properties are used together in the following combinations.

- **Disabled**—Compression is disabled if the level equals -1. The algorithm value is ignored when level equals -1.
- **Low compression**—Compression is enabled at a low level if level equals 0 and algorithm equals 10.
- **Medium compression**—Compression is enabled at a medium level if level equals 1 and algorithm equals 21.
- **High compression**—Compression is enabled at a high level if level equals 2 and algorithm equals 31.

ConnectionId

Returned by

Get-DtConnectionIds on page 35

Parameter of

Checkpoint-DtConnection on page 22, Get-DtBandwidthLimit on page 33, Get-DtSnapshot on page 68, Invoke-DtQueueTask on page 84, Remove-DtSnapshot on page 104, Resume-DtMirror on page 114, Resume-DtTarget on page 116, Set-DtBandwidthLimit on page 122, Start-DtMirror on page 145, Start-DtOrphansProcessing on page 147, Start-DtReplication on page 149, Start-DtVerify on page 151, Stop-DtMirror on page 155, Stop-DtReplication on page 157, Suspend-DtMirror on page 162, Suspend-DtTarget on page 164, Wait-DtMirrorComplete on page 179

Properties

Name	Type
Key	String
Value	Guid

ConnectionStartParameters

Parameter of

CoreConnectionOptions on page 220

Properties

Name	Type
ArchiveBinLocation	String
CompressionLevel	CompressionLevel on page 215
IsEncrypted	Boolean
IsMirrorEnabled	Boolean
IsPathBlockingEnabled	Boolean
IsReplicationEnabled	Boolean
IsRestore	Boolean
MirrorParameters	MirrorParameters on page 280
Schedule	Schedule on page 318
ScriptPoints	ScriptPoint [ ] on page 320
SnapshotSchedule	SnapshotSchedule on page 335

CoreConnectionDetails

Parameter of

FullServerJobDetails on page 256, JobStatistics on page 270

Properties

Name	Type
BandwidthCollar	Int32
CompressionEnabled	Boolean
CompressionLevel	Int32
ConnectionId	Int32
DiskQueueBytes	Int64
Encrypted	Boolean
InitialMirrorComplete	Boolean
ManagedConnectionId	Guid
MirrorBytesRemaining	Int64
MirrorBytesSent	Int64
MirrorBytesSkipped	Int64
MirrorBytesTransmitted	Int64
MirrorOpsQueued	Int64
MirrorPermillage	Int16
MirrorState	MirrorState on page 281
PeerMemoryLow	Boolean
ReplicationBytesQueued	Int64
ReplicationBytesSent	Int64
ReplicationBytesTransmitted	Int64
ReplicationOpsQueued	Int64
ReplicationState	ReplicationState on page 310
Restoring	Boolean
SourceAccessLevel	AccessLevel on page 186

Name	Type
SourceAvailable	Boolean
SourceClusterResourceState	ClusterResourceState on page 214
SourceEndpoint	String
SourceEngineAvailable	Boolean
SourceMachineName	String
SourceUniqueId	String
StartTime	DateTimeOffset
TargetAccessLevel	AccessLevel on page 186
TargetAvailable	Boolean
TargetEngineAvailable	Boolean
TargetLoaded	Boolean
TargetMachineName	String
TargetQueueBytes	Int64
TargetRoute	String
TargetState	TargetState on page 343
TargetUniqueId	String
TotalBytesSent	Int64
TotalBytesTransmitted	Int64
TotalOpsQueued	Int64
TransmissionMode	TransmissionMode on page 346

CoreConnectionOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
ConnectionStartParameters	ConnectionStartParameters on page 217
PathTransformations	PathTransformation on page 295
TargetAddress	String
TargetEnginePort	Int32

CoreMonitorDetails

Parameter of

JobStatistics on page 270

Properties

Name	Type
HighAvailabilityState	HighAvailabilityState on page 260
MonitoredAddressStatuses	MonitoredAddressStatus [ ] on page 284
MonitorId	Guid
MonitorName	String
RestoreStates	RestoreStates on page 315
TargetAvailable	Boolean

CoreMonitorOptions

Parameter of

FailbackOptions on page 241, JobOptions on page 268

Properties

Name	Type
MonitorConfiguration	MonitorConfiguration on page 282
ShouldPerformLanFailover	Boolean
TotalTimeAllowed	TimeSpan
UseTotalTimeAllowed	Boolean

CoreQualifcationResults

Parameter of

JobQualificationResults on page 269

Properties

Name	Type
CompressionLevels	CompressionLevel on page 215
DefaultAllToOneBasePath	String
SourceEnginePort	Int32
SourceIPAddresses	UnicastIPAddressInfo on page 347
SourceMachineName	String
SourceNetworkId	String
SourceNetworkInterfaces	NetworkInterfaceInfo on page 288
SourceProductInfo	ProductInfo on page 300
SourceVolumes	Volume on page 365
TargetEnginePort	Int32
TargetIPAddresses	UnicastIPAddressInfo on page 347
TargetMachineName	String
TargetNetworkInterfaces	NetworkInterfaceInfo on page 288
TargetProductInfo	ProductInfo on page 300
TargetVolumes	Volume on page 365

Credentials

Parameter of

ApplicationOptions on page 196, DnsDomainDetails on page 229, DnsOptions on page 230, EmailNotificationOptions on page 235, MonitorConfiguration on page 282, Server on page 322, V2VVirtualMachine on page 350

Properties

Name	Type
Domain	String
Password	String
UserName	String

CutoverDetails

Parameter of

FullServerJobDetails on page 256

Properties

Name	Type
PercentComplete	Int32
State	Int32

Datastore

Parameter of

VmHost on page 362

Properties

Name	Type
Capacity	Int32
FreeSpace	Int32
Name	String
RemoteId	String
Type	String
Url	String

DeleteOptions

Returned by

Get-DtUvraRecommendedRemoveOptions on page 74

Parameter of

Remove-DtJob on page 100

Properties

Name	Type
DeleteOnClusterResourceOffline	Boolean
DeleteReplica	Boolean
DiscardTargetQueue	Boolean

DesktopInteractionMode

Parameter of

ScriptPoint on page 320, Test-DtScript on page 170

Properties

Name	Enumeration
None	0
Interact	1

DnsDomainDetails

Parameter of

DnsOptions on page 230

Properties

Name	Type
Credentials	Credentials on page 224
DnsServers	DnsServerDetail on page 232
DomainName	String
IpAddressMappings	IpAddressMappings on page 264
ShouldUpdateTtl	Boolean
TtlValue	Int32

DnsOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
AdditionalDns	String
AlternateTrustee	String
Domains	DnsDomainDetails on page 229
Enabled	Boolean
SourceCredentials	Credentials on page 224
SourceServerInWorkgroup	Boolean
SourceServerName	String
TargetServerName	String

DnsRecordLock

Parameter of

ApplicationOptions on page 196

Properties

Name	Type
AdditionalDns	String
Addresses	String
DnsServer	String
Server	String
Trustees	String

DnsServerDetail

Parameter of

DnsDomainDetails on page 229

Properties

Name	Type
Address	String
Origin	String
SelectedForUpdate	Boolean

DTHVOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
IsWanFailoverEnabled	Boolean
OriginalSourceVM	VmInfo on page 363
ReplicaVMNetworkInterfaceInfo	VirtualNetworkInterfaceInfo on page 357
ReverseVHDMappings	VhdMapping on page 355
SelectedVirtualMachines	V2VVirtualMachine on page 350
SourceOsVersion	OperatingSystemVersion on page 292
SourceVM	VmInfo on page 363
SourceVMNetworkInterfaceInfo	VirtualNetworkInterfaceInfo on page 357
TargetOsVersion	OperatingSystemVersion on page 292
TargetVM	VmInfo on page 363
TotalSourceVHDSizeBytes	Int64
VirtualSwitchMapping	VirtualSwitchMapping on page 359
VLanMapping	VLanMapping on page 360

DTHVQualificationResults

Parameter of

JobQualificationResults on page 269

Properties

Name	Type
SourceVirtualSwitches	VirtualSwitchInfo [ ] on page 358
SourceVM	VMQualificationResults on page 364
TargetHost	ServerQualificationResults on page 326

EmailNotificationOptions

Returned by

Get-DtEmailNotificationOptions on page 37, Set-DtEmailNotificationOptions on page 124

Parameter of

Test-DtEmailNotification on page 168

Properties

Name	Type
ConnectionSecurity	SmtpConnectionSecurity on page 330
Enabled	Boolean
EntryIdFilter	String
EntryTypeFilter	String
From	String
IncludeEventDescriptionInSubject	Boolean
LoginToSmtpServer	Boolean
SmtpCredentials	Credentials on page 224
SmtpPort	Int32
SmtpServer	String
SubjectPrefix	String
To	String

EngineControlStatus

Parameter of

JobStatus on page 271

Properties

Name	Type
CanPauseMirror	Boolean
CanPauseTarget	Boolean
CanPauseTransmission	Boolean
CanProcessOrphans	Boolean
CanResumeMirror	Boolean
CanResumeTarget	Boolean
CanResumeTransmission	Boolean
CanSetBandwidth	Boolean
CanStartMirror	Boolean
CanStartTransmission	Boolean
CanStopMirror	Boolean
CanTakeSnapshot	Boolean
CanUpdateShares	Boolean
CanVerify	Boolean
ConnectionId	Guid
Role	String

EngineJobType

Parameter of

TargetStateInfo on page 344

Properties

Name	Enumeration
NormalJob	0
ImageJob	1
RecoveryJob	2
FullServerJob	4
GeoClusterJob	8
MigrationJob	16
FullServerRevertJob	32
VraRecoveryJob	64
VraMigrationJob	128
DataOnlyOption	256
VraJob	512
HyperVJob	1024
Win32MirrorOption	2048
SourceConnectionResourceJuob	4096
FullServerBackupJob	8192
UvraJob	16384
Invalid	65535

EventLogEntry

Returned by

Get-DtEventLogEntry on page 38

Properties

Name	Type
Category	String
CategoryNumber	Int16
EntryType	EventLogEntryType on page 239
Index	Int32
InstanceId	Int32
MachineName	String
Message	String
ReplacementStrings	String
Source	String
TimeGenerated	DateTime
TimeWritten	DateTime
UserName	String

EventLogEntryType

Parameter of

EventLogEntry on page 238

Properties

Name	Type
Error	String
Information	String
Warning	String

ExtendedLowLevelStates

Parameter of

JobStatus on page 271

Properties

Name	Type
Health	Health on page 259
MessageId	String
MessageIdFormatParameters	String

FailbackOptions

Parameter of

RecommendedFailbackOptions on page 303, Start-DtJobFailback on page 137

Properties

Name	Type
NewMonitorOptions	CoreMonitorOptions on page 222

FailoverDataAction

Parameter of

FailoverOptions on page 246, MonitorConfiguration on page 282

Properties

Name	Enumeration
Apply	0
Flush	1
Revert	2
Unknown	3

FailoverIPAddressesOption

Parameter of

MonitorConfiguration on page 282

Properties

Name	Enumeration
Monitored	0
All	1

FailoverItems

Parameter of

MonitorConfiguration on page 282

Properties

Name	Enumeration
None	0
IPAddresses	1
Name	2
Shares	4

FailoverMode

Parameter of

FailoverOptions on page 246

Properties

Name	Enumeration
Live	0
Test	1
Snapshot	2

FailoverOptions

Returned by

Get-DtUvraRecommendedFailoverOptions on page 70

Parameter of

RecommendedFailoverOptions on page 304, Start-DtJobFailover on page 139

Properties

Name	Type
FailoverDataAction	FailoverDataAction on page 242
FailoverMode	FailoverMode on page 245
FailoverType	FailoverType.Options on page 252
PerformTestFailover	Boolean
SnapshotId	Guid

FailoverProcessingOptions

Parameter of

MonitorConfiguration on page 282

Properties

Name	Enumeration
None	0
UserInterventionRequired	2
UseShareFile	1

FailoverReplaceActions

Parameter of

MonitorConfiguration on page 282

Properties

Name	Enumeration
None	0
Name	1
Address	2

FailoverScriptConfiguration

Parameter of

MonitorConfiguration on page 282

Properties

Name	Type
PostFailbackScript	String
PostFailbackScriptArgs	String
PostFailbackWait	Boolean
PostFailoverScript	String
PostFailoverScriptArgs	String
PostFailoverWait	Boolean
PreFailbackScript	String
PreFailbackScriptArgs	String
PreFailbackWait	Boolean
PreFailoverScript	String
PreFailoverScriptArgs	String
PreFailoverWait	Boolean
SourcePostFailbackScript	String
SourcePostFailbackScriptArgs	String

FailoverTrigger

Parameter of

MonitorConfiguration on page 282

Properties

Name	Enumeration
OneAddressFails	0
AllAddressesFail	1

FailoverType.Monitor

Parameter of

MonitorConfiguration on page 282

Properties

Name	Enumeration
Normal	0
FullServer	1

FailoverType.Options

Parameter of

FailoverOptions on page 246

Properties

Name	Enumeration
Manual	0
Automatic	1

Feature

Parameter of

JobInfo on page 266

Properties

Name	Type
Endpoint	Uri
Tag	String

FileSystemAttributes

Parameter of

LogicalVolume on page 274, PhysicalItem on page 296, Volume on page 365, VolumeOptions on page 367

Properties

Name	Enumeration
ReadOnly	1
Hidden	2
System	4
Directory	16
Archive	32
Normal	128
Temporary	256
SparseFile	512
ReparsePoint	1024
Compressed	2048
Offline	4096
NotContentIndexed	8192
Encrypted	16384

FullServerFailoverOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
AdditionalStagingFolders	String
CreateBackupConnection	Boolean
RemoveOrphans	Boolean
ShutdownSourceServer	Boolean
SourceChecksumAll	Boolean

FullServerJobDetails

Parameter of

JobStatistics on page 270

Properties

Name	Type
BackupConnectionDetails	CoreConnectionDetails on page 218
CutoverDetails	CutoverDetails on page 225
ProtectionConnectionDetails	CoreConnectionDetails on page 218
SystemVolumeRevertConnectionDetails	CoreConnectionDetails on page 218

FullServerNicMappings

Parameter of

SystemStateOptions on page 338

Properties

Name	Type
SourceNic	String
TargetNic	String
TargetNicList	String

Guid

Returned by

New-DtFilesAndFoldersJob on page 87, New-DtJob on page 89, New-DtWorkload on page 99

Parameter of

Add-DtPhysicalRule on page 18, Checkpoint-DtConnection on page 22, Close-DtWorkload on page 24, Confirm-DtJobOptions on page 25, Edit-DtJob on page 29, Get-DtBandwidthLimit on page 33, Get-DtConnectionIds on page 35, Get-DtJob on page 39, Get-DtJobActionStatus on page 41, Get-DtLogicalItem on page 43, Get-DtQualificationResults on page 52, Get-DtRecommendedFailbackOptions on page 54, Get-DtRecommendedFailoverOptions on page 56, Get-DtRecommendedPathTransform on page 61, Get-DtRecommendedRestoreOptions on page 62, Get-DtSnapshot on page 68, Get-DtUvraRecommendedFailoverOptions on page 70, Get-DtUvraRecommendedRemoveOptions on page 74, Get-DtWorkload on page 78, Get-DtWorkloadPhysicalItem on page 79, Invoke-DtQueueTask on page 84, New-DtFilesAndFoldersJob on page 87, New-DtJob on page 89, New-DtWorkload on page 99, Remove-DtJob on page 100, Remove-DtPhysicalRule on page 102, Remove-DtSnapshot on page 104, Repair-DtJobOptions on page 106, Resume-DtJob on page 112, Resume-DtMirror on page 114, Resume-DtTarget on page 116, Save-DtJobDiagnostics on page 118, Set-DtBandwidthLimit on page 122, Set-DtJobCredentials on page 125, Set-DtLogicalItemSelection on page 127, Start-DtJob on page 135, Start-DtJobFailback on page 137, Start-DtJobFailover on page 139, Start-DtJobRestore on page 141, Start-DtJobReverse on page 143, Start-DtMirror on page 145, Start-DtOrphansProcessing on page 147, Start-DtReplication on page 149, Start-DtVerify on page 151, Stop-DtJob on page 153, Stop-DtMirror on page 155, Stop-DtReplication on page 157, Suspend-DtJob on page 160, Suspend-DtMirror on page 162, Suspend-DtTarget on page 164, Undo-DtJobFailover on page 174, Update-DtShares on page 177, Wait-DtMirrorComplete on page 179

Properties

Name	Type
Guid	String

Health

Parameter of

ExtendedLowLevelStates on page 240, JobStatus on page 271

Properties

Name	Enumeration
Unknown	0
OK	1
Warning	2
Error	3

HighAvailabilityState

Parameter of

CoreMonitorDetails on page 221

Properties

Name	Enumeration
Illegal	-1
None	0
FailoverMonitoring	16
FailoverRequired	32
FailoverOccurring	48
FailbackRequired	64
FailbackOccurring	80
FailbackRemonitor	96

HighLevelState

Parameter of

JobStatus on page 271

Properties

Name	Enumeration
Unknown	0
Created	1
Deleting	2
FailedBack	3
FailedOver	4
FailingBack	5
FailingOver	6
FailoverFailed	7
FailoverPending	8
Mirroring	9
MirrorRequired	10
Paused	11
Pausing	12
Protecting	13
Provisioning	14
Restored	15
RestoreFailed	16
RestorePaused	17
RestoreRequired	18
Restoring	19
Resuming	20
Reversing	21
Reverting	22

Name	Enumeration
Starting	23
Stopped	24
Stopping	25
Undoing	26
RevertingSnapshot	27
UpdatingTargetImage	28
CredentialsRequired	29
ActivationCodeWarning	30
ActivationCodeError	31
EngineConnectionWarning	32
EngineConnectionError	33
EngineServiceWarning	34
EngineServiceError	35
ServerCommunicationWarning	36
ServerCommunicationError	37

InclusionMode

Parameter of

PhysicalRule on page 297

Properties

Name	Enumeration
Include	0
Exclude	1

IpAddressMappings

Parameter of

DnsDomainDetails on page 229

Properties

Name	Type
ShouldUpdateTtl	Boolean
SourceIP	String
TargetIP	String
TtlValue	Int32

JobAction

Parameter of

JobStatus on page 271

Properties

Name	Type
Duration	TimeSpan
ErrorCode	Int32
ExceptionMessage	String
Id	Guid
MessageFormatParameters	String
MessageId	String
RequestingUserName	String
Status	ActionStatus on page 187
Timestamp	DateTimeOffset
TitleFormatParameters	String
TitleId	String

JobInfo

Returned by

Get-DtJob on page 39

Parameter of

Checkpoint-DtConnection on page 22, Confirm-DtJobOptions on page 25, Edit-DtJob on page 29, Get-DtBandwidthLimit on page 33, Get-DtJob on page 39, Get-DtQualificationResults on page 52, Get-DtRecommendedFailbackOptions on page 54, Get-DtRecommendedFailoverOptions on page 56, Get-DtRecommendedRestoreOptions on page 62, Get-DtSnapshot on page 68, Get-DtUvraRecommendedFailoverOptions on page 70, Get-DtUvraRecommendedRemoveOptions on page 74, Invoke-DtQueueTask on page 84, Remove-DtJob on page 100, Remove-DtSnapshot on page 104, Resume-DtJob on page 112, Resume-DtMirror on page 114, Resume-DtTarget on page 116, Save-DtJobDiagnostics on page 118, Set-DtBandwidthLimit on page 122, Set-DtJobCredentials on page 125, Start-DtJob on page 135, Start-DtJobFailback on page 137, Start-DtJobFailover on page 139, Start-DtJobRestore on page 141, Start-DtJobReverse on page 143, Start-DtMirror on page 145, Start-DtOrphansProcessing on page 147, Start-DtReplication on page 149, Start-DtVerify on page 151, Stop-DtJob on page 153, Stop-DtMirror on page 155, Stop-DtReplication on page 157, Suspend-DtJob on page 160, Suspend-DtMirror on page 162, Suspend-DtTarget on page 164, Undo-DtJobFailover on page 174, Update-DtShares on page 177, Wait-DtMirrorComplete on page 179

Properties

Name	Type
CreatorUserName	String
Features	Feature [ ] on page 253
Id	Guid
JobPersistedState	N/A - Internal processing use only
JobType	String
LoadedFromDisk	Boolean
Managed	Boolean
Options	JobOptions on page 268
OtherHostUris	IDictionary
SourceHostUri	Uri
SourceUniqueId	String
Statistics	JobStatistics on page 270
Status	JobStatus on page 271

Name	Type
TargetHostUri	Uri
TargetUniqueId	String

JobOptions

Parameter of

Confirm-DtJobOptions on page 25, Edit-DtJob on page 29, JobInfo on page 266, New-DtFilesAndFoldersJob on page 87, New-DtJob on page 89, RecommendedJobOptions on page 306, RepairStatus on page 308

Properties

Name	Type
ApplicationOptions	ApplicationOptions on page 196
BandwidthOptions	BandwidthOptions on page 204
ClusterOptions	ClusterOptions on page 213
CoreConnectionOptions	CoreConnectionOptions on page 220
CoreMonitorOptions	CoreMonitorOptions on page 222
DnsOptions	DnsOptions on page 230
DtavOptions	Not applicable.This class is reserved for future features.
DTHVOptions	DTHVOptions on page 233
FullServerFailoverOptions	FullServerFailoverOptions on page 255
MonitoringOptions	MonitoringOptions on page 285
Name	String
SimpleFailoverMonitorOptions	SimpleFailoverMonitorOptions on page 329
SystemStateOptions	SystemStateOptions on page 338
TargetServicesOptions	TargetServicesOptions on page 340
UnmanagedConnectionOptions	UnmanagedConnectionOptions on page 348
VcdVappOptions	Not applicable.This class is reserved for future features.
VRAOptions	VRAOptions on page 369
Workload	Workload on page 373

JobQualificationResults

Returned by

Get-DtQualificationResults on page 52

Parameter of

RecommendedJobOptions on page 306

Properties

Name	Type
ClusterFilesAndFoldersQualificationResults	ClusterFilesAndFoldersQualifcationResults on page 212
CoreQualificationResults	CoreQualifcationResults on page 223
DtavQualificationResults	Not applicable.This class is reserved for future features.
DTHVQualificationResults	DTHVQualificationResults on page 234
SourceBehindNat	Boolean
SourceHostUri	Uri
SupportsFrameworkMonitoring	Boolean
SuppressFailoverMonitorOptions	Boolean
VcdVappQualificationResults	Not applicable.This class is reserved for future features.
VRAQualificationResults	VRAQualificationResults on page 370

JobStatistics

Parameter of

JobInfo on page 266

Properties

Name	Type
CoreConnectionDetails	CoreConnectionDetails on page 218
CoreMonitorDetails	CoreMonitorDetails on page 221
FullServerJobDetails	FullServerJobDetails on page 256

JobStatus

Parameter of

JobInfo on page 266

Properties

Name	Type
Actions	JobAction [ ] on page 265
CanDelete	Boolean
CanEdit	Boolean
CanFailback	Boolean
CanFailover	Boolean
CanPause	Boolean
CanRestore	Boolean
CanReverse	Boolean
CanStart	Boolean
CanStop	Boolean
CanUndoFailover	Boolean
EngineControlStatuses	EngineControlStatus on page 236
ExtendedLowLevelState	ExtendedLowLevelStates on page 240
Health	Health on page 259
HighLevelState	HighLevelState on page 261
IsInError	Boolean
LowLevelState	String
PermillageComplete	Int32
TargetState	String

LicenseType

Parameter of

ActivationCode on page 188

Properties

Name	Enumeration
NotApplicable	0
Limited	1
Single	2
Site	3
NodeLockedKey	4

LogicalItems

Returned by

Get-DtLogicalItem on page 43

Parameter of

Get-DtLogicalItem on page 43

Properties

Name	Type
IsContainer	Boolean
IsReadOnly	Boolean
ItemType	String
Metadata	String
Name	String
Path	String
Saturation	SaturationLevel on page 317
SaturationDefault	SaturationLevel on page 317

LogicalVolume

Parameter of

VolumeGroup on page 366

Properties

Name	Type
Attributes	FileSystemAttributes on page 254
AvailableFreeSpace	Int64
CreationTime	DateTime
DesiredSize	Int64
DiskControllerType	String
DiskProvisoningType	String
DriveFormat	String
DriveType	DriveType
IsContainer	Boolean
IsReadOnly	Boolean
IsSystemDrive	Boolean
ItemType	String
Label	String
LastAccessTime	DateTime
LastWriteTime	DateTime
LogicalVolumeName	String
Metadata	String
Name	String
Path	String
Saturation	SaturationLevel on page 317
Size	Int64
TotalSize	Int64
VirtualDiskPath	String

Name	Type
VolumeSignature	Int16
VolumeType	String

LogMessage

Returned by

Get-DtLogMessage on page 44

Properties

Name	Type
Hash	Int32
Id	Int32
MessageType	String
ProcessId	Int32
Sequence	Int32
Source	String
Text	String
ThreadId	Int32
Timestamp	DateTime
TimeStampOffset	DateTimeOffset

LvmOptions

Parameter of

ServerInfo on page 324, VRAOptions on page 369

Properties

Name	Type
VolumeGroup	VolumeGroup [ ] on page 366

MirrorComparisonCriteria

Parameter of

MirrorParameters on page 280, RestoreParameters on page 313, VerifySchedule on page 353

Properties

Name	Enumeration
None	0
Newer	1
Checksum	2

MirrorOperationOptions

Parameter of

MirrorParameters on page 280, VerifySchedule on page 353

Properties

Name	Enumeration
None	0
Synchronize	1
Report	2
CalculateDifferences	4
CalculateSize	8
ProcessOrphans	16

MirrorParameters

Parameter of

ConnectionStartParameters on page 217, Start-DtMirror on page 145

Properties

Name	Type
ComparisonCriteria	MirrorComparisonCriteria on page 278
Options	MirrorOperationOptions on page 279
OverrideJobOrphansProcessing	Boolean

MirrorState

Returned by

Wait-DtMirrorComplete on page 179

Parameter of

CoreConnectionDetails on page 218

Properties

Name	Enumeration
Calculating	0
Idle	1
Mirror	2
Pause	3
RemoveOrphans	4
RepsetVerify	5
Restore	6
Stopped	7
Waiting	8
Unknown	9

MonitorConfiguration

Parameter of

CoreMonitorOptions on page 222

Properties

Name	Type
ActiveDirectoryCredentials	Credentials on page 224
ActiveDirectoryOptions	ActiveDirectoryOptions on page 192
Addresses	MonitoredAddressConfiguration [ ] on page 283
DataAction	FailoverDataAction on page 242
FailoverIPAddressOption	FailoverIPAddressesOption on page 243
FailoverType	FailoverType.Monitor on page 251
ItemsToFailover	FailoverItems on page 244
MaxScriptFailures	Int32
Name	String
ProcessingOptions	FailoverProcessingOptions on page 247
ReplaceActions	FailoverReplaceActions on page 248
ScriptMonitorEngine	String
ScriptMonitorName	String
Scripts	FailoverScriptConfiguration on page 249
SourceEndpoint	String
SSMLogPath	String
SSMManualReboot	Boolean
SSMRecoveryType	SSMRecoveryType on page 336
SSMSourceNicGuids	String
SSMStagingPath	String
SSMTargetNicGuids	String
Trigger	FailoverTrigger on page 250

MonitoredAddressConfiguration

Parameter of

MonitorConfiguration on page 282

Properties

Name	Type
Address	String
MacAddress	String
MaxPingAttempts	Int16
NicName	String
PingInterval	TimeSpan
PingMethods	PingMethods on page 299
SubnetMask	String

MonitoredAddressStatus

Parameter of

CoreMonitorDetails on page 221

Properties

Name	Type
Address	String
Alive	Boolean
FailoverConditionMet	Boolean
RemainingTime	TimeSpan
WarningConditionMet	Boolean

MonitoringOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
ServiceMonitoringEnabled	Boolean
ServiceMonitoringOptions	ServiceMonitoringOptions on page 328

Network

Parameter of

VmHost on page 362

Properties

Name	Type
IsDistributed	Boolean
Name	String
PortgroupKey	String

NetworkAdaptersInfo

Parameter of

VMQualificationResults on page 364

Properties

Name	Type
SwitchPort	SwitchPortInfo on page 337
VirtualSwitch	VirtualSwitchInfo on page 358

NetworkInterfaceInfo

Parameter of

CoreQualifcationResults on page 223, ServerInfo on page 324

Properties

Name	Type
Description	String
DnsDomain	String
DnsServers	String
Gateways	String
Guid	String
Index	Int32
InterfaceIndex	Int32
IPAddresses	UnicastIPAddressInfo [ ] on page 347
MacAddresses	String
Name	String
PnpInstanceId	String
ServiceName	String

OperatingSystemArchitecture

Parameter of

OperatingSystemInfo on page 290

Properties

Name	Enumeration
x86	0
ia64	6
x64	9

OperatingSystemInfo

Parameter of

ServerInfo on page 324

Properties

Name	Type
Architecture	OperatingSystemArchitecture on page 289
CSDVersion	Int32
ProductSuite	Int32
ProductType	OperatingSystemProductType on page 291
ServicePack	String
Version	OperatingSystemVersion on page 292
VersionString	String

OperatingSystemProductType

Parameter of

OperatingSystemInfo on page 290

Properties

Name	Enumeration
None	0
Workstation	1
DomainController	2
Server	3

OperatingSystemVersion

Parameter of

DTHVOptions on page 233, OperatingSystemInfo on page 290

Properties

Name	Type
Build	Int32
Major	Int32
Minor	Int32
Revision	Int32

OrphansSchedule

Parameter of

Schedule on page 318

Properties

Name	Type
IsEnabled	Boolean

PathBlocking

Returned by

Get-DtPathBlocking on page 48

Properties

Name	Type
BlockingMode	BlockingMode on page 210
Paths	String
SourceAddress	String

PathTransformation

Returned by

Get-DtRecommendedPathTransform on page 61

Parameter of

CoreConnectionOptions on page 220, RestoreParameters on page 313

Properties

Name	Type
SourcePath	String
TargetPath	String

PhysicalItem

Returned by

Get-DtPhysicalItem on page 49, Get-DtWorkloadPhysicalItem on page 79

Parameter of

ChangedItems on page 211, Get-DtPhysicalItem on page 49, Get-DtWorkloadPhysicalItem on page 79

Properties

Name	Type
Attributes	FileSystemAttributes on page 254
CreationTime	DateTime
IsContainer	Boolean
IsReadOnly	Boolean
ItemType	String
LastAccessTime	DateTime
LastWriteTime	DateTime
Metadata	String
Name	String
Path	String
Saturation	SaturationLevel on page 317
Size	Int64

PhysicalRule

Parameter of

ChangedItems on page 211, Workload on page 373

Properties

Name	Type
Inclusion	InclusionMode on page 263
IsReadOnly	Boolean
Metadata	String
Path	String
Recursion	RecursionMode on page 307

PhysicalVolume

Parameter of

VolumeGroup on page 366

Properties

Name	Type
Attributes	FileSystemAttributes on page 254
AvailableFreeSpace	Int64
CreationTime	DateTime
DesiredSize	Int64
DiskControllerType	String
DiskProvisoningType	String
DriveFormat	String
DriveType	DriveType
IsContainer	Boolean
IsReadOnly	Boolean
IsSystemDrive	Boolean
ItemType	String
Label	String
LastAccessTime	DateTime
LastWriteTime	DateTime
Metadata	String
Name	String
Path	String
Saturation	SaturationLevel on page 317
Size	Int64
TotalSize	Int64
VirtualDiskPath	String
VolumeSignature	Int16
VolumeType	String

PingMethods

Parameter of

MonitoredAddressConfiguration on page 283

Properties

Name	Enumeration
None	0
Network	1
Service	2
Manual	4
Script	8
ForceUpdate	256

ProductInfo

Returned by

CoreQualifcationResults on page 223, Get-DtProductInfo on page 50

Properties

Name	Type
ActivationStatus	ActivationStatus on page 191
CanPauseTarget	Boolean
CanResumeTarget	Boolean
EgineModuleStatus	Int32
EnginePort	Int32
GatewaySessionKey	Int64
InstallationPath	String
LocalEndpoints	String
MachineName	String
ManagementServiceVersion	ProductVersion on page 301
Name	String
NatEndpoints	String
ReservedAddress	String
UniqueID	String
Version	ProductVersion on page 301

ProductVersion

Parameter of

ProductInfo on page 300, TargetStateInfo on page 344

Properties

Name	Type
Build	Int32
Hotfix	Int32
Major	Int32
Minor	Int32
ServicePack	Int32

PSCredential

Parameter of

DnsDomainDetails on page 229, DnsOptions on page 230, EmailNotificationOptions on page 235, New-DtServer on page 92, New-DtUri on page 95, New-DtUvraServer on page 97, Server on page 322, Set-DtJobCredentials on page 125, Set-DtScriptCredentials on page 132, Set-DtServerCredential on page 134, Test-DtActiveDirectoryCredentials on page 166, Test-DtScriptCredentials on page 172,

Properties

Name	Type
UserName	String
Password	SecureString

RecommendedFailbackOptions

Returned by

Get-DtRecommendedFailbackOptions on page 54

Properties

Name	Type
FailbackOptions	FailbackOptions on page 241
IsSourceNew	Boolean
RestoreStatus	RestoreStatus on page 316

RecommendedFailoverOptions

Returned by

Get-DtRecommendedFailoverOptions on page 56

Properties

Name	Type
FailoverOptions	FailoverOptions on page 246
IsTestFailoverSupported	Boolean
Snapshots	SnapshotEntry [ ] on page 333
WarningTextTestFailover	Boolean
WarnUserOfInconsistentProtectionData	Boolean

RecommendedRestoreOptions

Returned by

Get-DtRecommendedRestoreOptions on page 62

Properties

Name	Type
CanClearRestoreRequired	Boolean
PossibleSourceAddresses	String
RestoreOptions	RestoreOptions on page 312
SameSourceOnly	Boolean

RecommendedJobOptions

Returned by

Get-DtRecommendedJobOptions on page 58, Get-DtUvraRecommendedJobOptions on page 72

Properties

Name	Type
JobOptions	JobOptions on page 268
JobQualificationResults	JobQualificationResults on page 269

RecursionMode

Parameter of

PhysicalRule on page 297

Properties

Name	Enumeration
Recursive	0
NonRecursive	1

RepairStatus

Returned by

Get-DtRepairJobOptionsStatus on page 64

Properties

Name	Type
Task	ActivityStatusEntry on page 194
JobOptions	JobOptions on page 268

ReplicationSetUsageType

Parameter of

TargetStateInfo on page 344

Properties

Name	Enumeration
Invalid	-1
Normal	0
SystemState	1
GeoCluster	2

ReplicationState

Parameter of

CoreConnectionDetails on page 218

Properties

Name	Enumeration
NotReplicating	0
OutOfMemory	1
Pending	2
Replicating	3
Unknown	4
Watchdog	5
Ready	6

ReplicaVmInfo

Parameter of

VRAOptions on page 369

Properties

Name	Type
Address	String
BootVolumeSignature	Int8
Cpus	Int32
DisplayName	String
GuestOS	String
GuestUri	Uri
Id	Guid
Memory	Int64
OperatingSystem	String
Path	String
PrestageFolder	String
RunOnceAtStartup	String
SnapshotDataPath	String
SnapshotFileName	String
SystemDirectory	String
VirtualHardDiskPath	String

RestoreOptions

Parameter of

RecommendedRestoreOptions on page 305

Properties

Name	Type
ClearRestoreRequired	Boolean
RestoreParameters	RestoreParameters on page 313
RestoreTargetHostUri	Uri

RestoreParameters

Parameter of

RestoreOptions on page 312

Properties

Name	Type
ArchiveBinLocation	String
ArchiveOption	ArchiveOption on page 198
CompressionLevel	CompressionLevel on page 215
MirrorComparisonCritera	MirrorComparisonCriteria on page 278
OriginalSourceName	String
OriginalTargetRoute	String
PathTransformations	PathTransformation on page 295
ProcessOrphans	Boolean
ReplicationSetName	String
RestoreOptions	RestoreParametersRestoreOptions on page 314

RestoreParametersRestoreOptions

Parameter of

RestoreParameters on page 313

Properties

Name	Enumeration
None	0
UseTargetWorkload	1
RestoreWorkloadToSource	2
OverwriteExistingFiles	4

RestoreStates

Parameter of

CoreMonitorDetails on page 221

Properties

Name	Enumeration
None	0
OldServer	1
Required	2
Connected	4
MultiConnect	8
Mirroring	16
MirrorStopped	32
OpDropped	64
OpRetrying	128

RestoreStatus

Parameter of

RecommendedFailbackOptions on page 303

Properties

Name	Enumeration
NotStarted	0
Restoring	1
Restored	2

SaturationLevel

Parameter of

LogicalItems on page 273, LogicalVolume on page 274, PhysicalItem on page 296, Volume on page 365

Properties

Name	Enumeration
Unknown	0
None	1
Partial	2
Full	3
Error	4

Schedule

Parameter of

ConnectionStartParameters on page 217

Properties

Name	Type
Archive	ArchiveSchedule on page 199
Bandwidth	BandwidthSchedule on page 205
Orphans	OrphansSchedule on page 293
Transmission	This property is no longer supported.
Verify	VerifySchedule on page 353

ScriptExecutionMode

Parameter of

ScriptPoint on page 320

Properties

Name	Enumeration
Synchronous	0
Asynchronous	1

ScriptPoint

Parameter of

ConnectionStartParameters on page 217

Properties

Name	Type
Arguments	String
ExecutionMode	ScriptExecutionMode on page 319
InteractionMode	DesktopInteractionMode on page 228
Path	String
Type	ScriptPointType on page 321

ScriptPointType

Parameter of

ScriptPoint on page 320

Properties

Name	Enumeration
MirrorStart	0
MirrorComplete	1
MirrorStop	2

Server

Returned by

New-DtServer on page 92, New-DtUvraServer on page 97, Set-DtServerCredential on page 134

Parameter of

Add-DtPhysicalRule on page 18, Add-DtUvraPhysicalRule on page 20, Checkpoint-DtConnection on page 22, Close-DtWorkload on page 24, Confirm-DtJobOptions on page 25, Disconnect-DtServer on page 28, Edit-DtJob on page 29, Get-DtAccessLevel on page 31, Get-DtActivationStatus on page 32, Get-DtBandwidthLimit on page 33, Get-DtConnectionIds on page 35, Get-DtDiagnostics on page 36, Get-DtEmailNotificationOptions on page 37, Get-DtEventLogEntry on page 38, Get-DtJob on page 39, Get-DtJobActionStatus on page 41, Get-DtLogicalItem on page 43, Get-DtLogMessage on page 44, Get-DtOption on page 47, Get-DtPathBlocking on page 48, Get-DtPhysicalItem on page 49, Get-DtProductInfo on page 50, Get-DtProxiedServerInfo on page 51, Get-DtQualificationResults on page 52, Get-DtRecommendedFailbackOptions on page 54, Get-DtRecommendedFailoverOptions on page 56, Get-DtRecommendedJobOptions on page 58, Get-DtRecommendedPathTransform on page 61, Get-DtRecommendedRestoreOptions on page 62, Get-DtScriptCredentials on page 66, Get-DtServerInfo on page 67, Get-DtSnapshot on page 68, Get-DtUvraRecommendedFailoverOptions on page 70, Get-DtUvraRecommendedJobOptions on page 72, Get-DtUvraRecommendedRemoveOptions on page 74, Get-DtUvraVmHost on page 76, Get-DtVerificationStatus on page 77, Get-DtWorkload on page 78, Get-DtWorkloadPhysicalItem on page 79, Get-DtWorkloadType on page 80, Invoke-DtQueueTask on page 84, New-DtFilesAndFoldersJob on page 87, New-DtJob on page 89, New-DtTaskParameters on page 94, New-DtWorkload on page 99, Remove-DtJob on page 100, Remove-DtPhysicalRule on page 102, Remove-DtSnapshot on page 104, Repair-DtJobOptions on page 106, Restart-DtReplicationService on page 111, Resume-DtJob on page 112, Resume-DtMirror on page 114, Resume-DtTarget on page 116, Save-DtJobDiagnostics on page 118, Set-DtActivationCode on page 120, Set-DtBandwidthLimit on page 122, Set-DtEmailNotificationOptions on page 124, Set-DtJobCredentials on page 125, Set-DtLogicalItemSelection on page 127, Set-DtOption on page 129, Set-DtPathBlocking on page 131, Set-DtScriptCredentials on page 132, Start-DtJob on page 135, Start-DtJobFailback on page 137, Start-DtJobFailover on page 139, Start-DtJobRestore on page 141, Start-DtJobReverse on page 143, Start-DtMirror on page 145, Start-DtOrphansProcessing on page 147, Start-DtReplication on page 149, Start-DtVerify on page 151, Stop-DtJob on page 153, Stop-DtMirror on page 155, Stop-DtReplication on page 157, Stop-DtReplicationService on page 159, Suspend-DtJob on page 160, Suspend-DtMirror on page 162, Suspend-DtTarget on page 164, Test-DtActiveDirectoryCredentials on page 166, Test-DtEmailNotification on page 168, Test-DtScript on page 170, Test-DtScriptCredentials on page 172, Undo-DtJobFailover on page 174, Update-DtShares on page 177, Wait-DtMirrorComplete on page 179

Properties

Name	Type
Credentials	Credentials on page 224
HostName	String
Port	Int32
Role	String

ServerActivationInformation

Returned by

Get-DtOnlineActivationRequest on page 46

Properties

Name	Type
Code	String
ServerInformation	String
ServerName	String

ServerInfo

Returned by

Get-DtProxiedServerInfo on page 51, Get-DtServerInfo on page 67

Parameter of

Get-DtUvraRecommendedJobOptions on page 72

Properties

Name	Type
BiosGuid	Guid
BootVolume	String
Domain	String
FullyQualifiedDomain	String
HalInternalName	String
HalVersion	String
IsClustered	Boolean
IsHostedByHyperV	Boolean
IsHostedByVMware	Boolean
IsHostedByXen	Boolean
IsHyperVHost	Boolean
IsReplicationEnabled	Boolean
IsSBS	Boolean
IsSSE	Boolean
LvmOptions	LvmOptions on page 277
ManagementPort	Int32
MemrorySize	Int64
Name	String
NetworkInterfaces	NetworkInterfaceInfo [ ] on page 288
NodeLockedServerInfo	String
OperatingSystem	OperatingSystemInfo on page 290

Name	Type
ProcessorCount	Int32
ProgramFilesPath	String
SystemPath	String
SystemRoot	String
SystemStateDefinition	String
SystemVolume	String
Volumes	Volume [ ] on page 365

ServerQualificationResults

Parameter of

DTHVQualificationResults on page 234, VRAQualificationResults on page 370

Properties

Name	Type
Cpus	Int32
Memory	Int64
Version	String
VirtualSwitches	VirtualSwitchInfo [ ] on page 358
Volumes	VolumeQualificationResults on page 368

ServiceInformation

Parameter of

ApplicationOptions on page 196, ServiceMonitoringOptions on page 328, TargetServicesOptions on page 340

Properties

Name	Type
DisplayName	String
Name	String
Selected	Boolean

ServiceMonitoringOptions

Parameter of

MonitoringOptions on page 285

Properties

Name	Type
RepeatCount	Int32
Services	ServiceInformation on page 327
StartService	Boolean

SimpleFailoverMonitorOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
IsUserInterventionRequired	Boolean
MaxPingAttempts	Int16
PingInterval	TimeSpan
TimeAllowed	TimeSpan
UseTimeAllowed	Boolean

SmtpConnectionSecurity

Parameter of

EmailNotificationOptions on page 235

Properties

Name	Enumeration
Plain	0
SSL	1
TLS	2

SnapshotAge

Parameter of

Get-DtSnapshot on page 68

Properties

Name	Enumeration
Old	0
All	1

SnapshotCreationReason

Parameter of

SnapshotEntry on page 333

Properties

Name	Enumeration
Manual	0
Automatic	1
Scheduled	2
Deferred	3
DataTest	4

SnapshotEntry

Returned by

Get-DtSnapshot on page 68

Parameter of

RecommendedFailoverOptions on page 304, TargetStateInfo on page 344

Properties

Name	Type
Comment	String
Id	Guid
Reason	SnapshotCreationReason on page 332
States	TargetState on page 343
Timestamp	DateTimeOffset

SnapshotQuality

Parameter of

Get-DtSnapshot on page 68

Properties

Name	Enumeration
Good	0
Bad	1

SnapshotSchedule

Parameter of

ConnectionStartParameters on page 217

Properties

Name	Type
Interval	TimeSpan
IsEnabled	Boolean
MaxNumberOfSnapshots	Int32
StartTime	DateTime

SSMRecoveryType

Parameter of

MonitorConfiguration on page 282

Properties

Name	Enumeration
LAN	0
WAN	1

SwitchPortInfo

Parameter of

NetworkAdaptersInfo on page 287, VLanMapping on page 360

Properties

Name	Type
AccessVLanId	Int32
Guid	String
Name	String

SystemStateOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
AlternateVolumeMapping	String
AlternateVolumeStaging	Boolean
ClearMonitor	Boolean
IsWanFailover	Boolean
KeepTargetActivationCode	Boolean
NicMappings	FullServerNicMappings on page 257
ServicesToStopOptions	TargetServicesToStop on page 342
ShouldApplyDiskSignatures	Boolean
SourceReservedAddress	String
StagingFolder	String
TargetReservedAddress	String

TargetFileServerQualificationResults

Parameter of

ClusterFilesAndFoldersQualifcationResults on page 212

Properties

Name	Type
ClusterResourceGroupIPAddresses	UnicastIPAddressInfo on page 347
ClusterResourceGroupName	String
CurrentOwnerNodeName	String
DiskSize	Int64
DriveLetter	String
FreeSpace	Int64
IsVolumeCSV	Boolean
RecommendedGroup	Boolean

TargetServicesOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
FailoverServices	ServiceInformation on page 327
StartAndStopServices	Boolean

TargetServiceStatus

Parameter of

TargetServicesToStop on page 342

Properties

Name	Enumeration
Stopped	0
Started	1
StopPending	2
StartPending	3
ResumePending	4
PausePending	5
Paused	6
Unknown	7

TargetServicesToStop

Parameter of

SystemStateOptions on page 338

Properties

Name	Type
Failover	Boolean
IsCritical	Boolean
KeepRunningNonCritical	Boolean
ServiceDescription	String
ServiceName	String
State	TargetServiceStatus on page 341

TargetState

Parameter of

CoreConnectionDetails on page 218, SnapshotEntry on page 333, TargetStateInfo on page 344

Properties

Name	Enumeration
Good	0
Mirroring	1
MirrorStopped	2
OpDropped	4
Retrying	8
Paused	16
PausePending	32
RestoreRequired	64
ReplicationPending	128
SnapshotReverted	256
FailoverUnblocked	512
Disconnected	1024
SroImage	2048
FailoverMonitoring	4096
TransactionsPending	8192
GCReplicationComplete	16384
MarkedForDeletion	32768
TargetPathBlocked	2147483648
Unknown	4294967296

TargetStateInfo

Parameter of

UnmanagedConnectionOptions on page 348

Properties

Name	Type
ConnectionId	Guid
ConnectTime	DateTimeOffset
EngineConnectionId	Int32
EngineJobType	EngineJobType on page 237
HasSnapshotSchedule	Boolean
LastUpdateTime	DateTimeOffset
NextScheduledSnapshot	DateTimeOffset
Paths	String
QueueBytes	Int64
ReplicationSetName	String
ReplicationSetUsageType	ReplicationSetUsageType on page 309
ScheduledSnapshotInterval	TimeSpan
Snapshots	SnapshotEntry on page 333
SourceEndpoint	String
SourceEndpointFromSource	String
SourceMachineName	String
SourceVersion	ProductVersion on page 301
TargetEndpoint	String
TargetStates	TargetState on page 343

TaskParameters

Returned by

New-DtTaskParameters on page 94

Parameter of

Invoke-DtQueueTask on page 84

Properties

Name	Type
Arguments	String
Script	String

TransmissionMode

Parameter of

CoreConnectionDetails on page 218

Properties

Name	Enumeration
Error	0
Paused	1
Started	2
Scheduled	3
Stopped	4
Unknown	5

UnicastIPAddressInfo

Parameter of

CoreQualifcationResults on page 223, NetworkInterfaceInfo on page 288, TargetFileServerQualificationResults on page 339, VirtualNetworkInterfaceInfo on page 357

Properties

Name	Type
IPAddress	String
IPv4Mask	String
IsDHCP	Boolean
IsNAT	Boolean
IsOnline	Boolean

UnmanagedConnectionOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
TargetStateInfo	TargetStateInfo on page 344

V2VQualificationResults

Parameter of

VRAQualificationResults on page 370

Properties

Name	Type
ReverseNetworkRoutes	String
SourceActivationCodeInfo	ActivationCodeInfo on page 189
SourceEnginePort	Int32
SourceHostAllVirtualMachines	VmInfo on page 363
SourceHostVirtualSwitches	VirtualSwitchInfo [ ] on page 358
SourceHyperVHostSelectedVirtualMachines	V2VVirtualMachine on page 350

V2VVirtualMachine

Parameter of

DTHVOptions on page 233, V2VQualificationResults on page 349

Properties

Name	Type
ActivationCode	String
BiosGuid	Guid
CPUs	Int32
Credentials	Credentials on page 224
DiskFormat	String
DisplayName	String
GuestOS	String
Guid	Guid
Memory	Int64
NetworkId	String
NetworkInteraceInfo	VirtualNetworkInterfaceInfo on page 357
Path	String
SnapshotPath	String
SystemDirectory	String
TargetPath	String
Volumes	VolumeOptions on page 367

VerificationStatus

Returned by

Get-DtVerificationStatus on page 77

Properties

Name	Type
Step	VerificationStep [ ] on page 352
Task	ActivityStatusEntry on page 194

VerificationStep

Parameter of

Repair-DtJobOptions on page 106, VerificationStatus on page 351

Properties

Name	Type
CanFix	Boolean
Id	String
Index	Int32
MessageFormatParameters	String
MessageKey	String
Progress	Int32
Status	ActivityCompletionStatus on page 193
TitleFormatParameters	String [ ]
TitleKey	String

VerifySchedule

Parameter of

Schedule on page 318

Properties

Name	Type
Criteria	MirrorComparisonCriteria on page 278
Interval	TimeSpan
IsEnabled	Boolean
Options	MirrorOperationOptions on page 279
StartTime	DateTime

VHDInfo

Parameter of

VMQualificationResults on page 364

Properties

Name	Type
Drive	String
FileSize	Int64
MaxInternalSize	Int64
ParentPath	String
Path	String
Type	Int32

VhdMapping

Parameter of

DTHVOptions on page 233

Properties

Name	Type
SourceVHD	String
TargetVHD	String

VirtualMachinePowerState

Parameter of

Vm on page 361

Properties

Name	Enumeration
PoweredOff	0
PoweredOn	1
Suspended	2

VirtualNetworkInterfaceInfo

Parameter of

DTHVOptions on page 233, V2VVirtualMachine on page 350, VRAOptions on page 369

Properties

Name	Type
Description	String
DnsDomain	String
DnsServers	String
Gateways	String
Guid	String
Index	Int32
InterfaceIndex	Int32
IPAddresses	UnicastIPAddressInfo on page 347
MacAddresses	String
Name	String
PnpInstanceId	String
ServiceName	String
VirtualNetwork	String
VirtualNictype	String

VirtualSwitchInfo

Parameter of

DTHVQualificationResults on page 234, NetworkAdaptersInfo on page 287, ServerQualificationResults on page 326, V2VQualificationResults on page 349, VirtualSwitchMapping on page 359

Properties

Name	Type
Label	String
SwitchUuid	String

VirtualSwitchMapping

Parameter of

DTHVOptions on page 233

Properties

Name	Type
SourceVirtualSwitch	VirtualSwitchInfo on page 358
TargetVirtualSwitch	VirtualSwitchInfo on page 358

VLanMapping

Parameter of

DTHVOptions on page 233

Properties

Name	Type
SourceSwitchPort	SwitchPortInfo on page 337
TargetSwitchPort	SwitchPortInfo on page 337

Vm

Parameter of

VmHost on page 362

Properties

Name	Type
GuestFullName	String
GuestId	String
HostName	String
IsTemplate	Boolean
MemoryMB	Int32
Name	String
NumCPU	Int32
PowerState	VirtualMachinePowerState on page 356
Uuid	String

VmHost

Returned by

Get-DtUvraVmHost on page 76

Parameter of

Get-DtUvraRecommendedJobOptions on page 72

Properties

Name	Type
BuildNum	String
CpuMHz	Int32
CpuUsageMHz	Int32
Datastores	Datastore on page 226
MemoryMB	Int32
MemoryUsageMB	Int32
Model	String
Name	String
Networks	Network on page 286
NumOfCpu	Int16
NumOfNics	Int32
ProcessorType	String
Uuid	String
Vendor	String
Version	String
VirtualMachines	Vm on page 361

VmInfo

Parameter of

DTHVOptions on page 233, V2VQualificationResults on page 349, VRAOptions on page 369

Properties

Name	Type
Address	String
BootVolumeSignature	Int8
DisplayName	String
GuestOS	String
GuestUri	Uri
Id	Guid
Path	String
SnapshotDataPath	String
SnapshotFileNames	String
SystemDirectory	String
VirtualHardDiskPath	String

VMQualificationResults

Parameter of

DTHVQualificationResults on page 234

Properties

Name	Type
Address	String
DisplayName	String
IsHeartbeatInstalled	Boolean
NetworkAdapters	NetworkAdaptersInfo on page 287

Volume

Parameter of

CoreQualifcationResults on page 223, ServerInfo on page 324

Properties

Name	Type
Attributes	FileSystemAttributes on page 254
AvailableFreeSpace	Int64
CreationTime	DateTime
DriveFormat	String
DriveType	DriveType
IsContainer	Boolean
IsReadOnly	Boolean
IsSystemDrive	Boolean
ItemType	String
Label	String
LastAccessTime	DateTime
LastWriteTime	DateTime
Metadata	String
Name	String
Path	String
Saturation	SaturationLevel on page 317
Size	Int32
TotalSize	Int64
VolumeType	String

VolumeGroup

Parameter of

LvmOptions on page 277

Properties

Name	Type
LogicalVolume	LogicalVolume [ ] on page 274
MaxPhysicalVolumeSize	Int64
Name	String
PhysicalVolume	PhysicalVolume [ ] on page 298
SourceVolumeGroupSize	Int64

VolumeOptions

Parameter of

V2VVirtualMachine on page 350, VRAOptions on page 369

Properties

Name	Type
Attributes	FileSystemAttributes on page 254
AvailableFreeSpace	Int32
CreationTime	DateTime
DesiredSize	Int64
DiskControllerType	String
DiskProvisioningType	String
DriveFormat	String
DriveType	DriveType
IsContainer	Boolean
IsReadOnly	Boolean
IsSystemDrive	Boolean
ItemType	String
Label	String
LastAccessTime	DateTime
LastWriteTime	DateTime
Metadata	String
Name	String
Path	String
Saturation	SaturationLevel on page 317
Size	Int32
TotalSize	Int32
VirtualDiskPath	String
VolumeSignature	Int8

VolumeQualificationResults

Parameter of

ServerQualificationResults on page 326

Properties

Name	Type
ClusterResourceGroupName	String
CurrentOwnerNodeName	String
DiskSize	Int64
DriveLetter	String
FreeSpace	Int64
IsSystemVolume	Boolean
IsVolumeCSV	Boolean
MaxFileSize	Int64
ProvisionedSpace	Int64
Url	String

VRAOptions

Parameter of

JobOptions on page 268

Properties

Name	Type
Hpervisor	String
IsSourceHostCluster	Boolean
IsWanFailoverEnabled	Boolean
LvmOptions	LvmOptions on page 277
ReplicaApplianceInfo	VmInfo on page 363
ReplicaESXHostName	String
ReplicaNetworkInterfaceInfo	VirtualNetworkInterfaceInfo on page 357
ReplicaVmInfo	ReplicaVmInfo on page 311
ReverseCount	Int32
ReverseRoute	String
SourceApplianceInfo	VmInfo on page 363
SourceESXHostName	String
SourceNetworkInterfaceInfo	VirtualNetworkInterfaceInfo on page 357
SourceProductLicense	String
SourceVmInfo	VmInfo on page 363
VirtualSwitchMapping	VirtualSwitchMapping on page 359
Volumes	VolumeOptions on page 367
WorkloadCustomizationOptions	VRAWorkloadCustomizationOptions on page 371

VRAQualificationResults

Parameter of

JobQualificationResults on page 269

Properties

Name	Type
PreexistingDisksFileName	String
SourceServerMemorySize	Int64
SourceServerProcessorCount	Int32
TargetHost	ServerQualificationResults on page 326
V2VQualificationResults	V2VQualificationResults on page 349

VRAWorkloadCustomizationOptions

Parameter of

VRAOptions on page 369

Properties

Name	Type
NoReplication	Boolean
PowerupReplicaAfterFailover	Boolean
ShouldShutdownSource	Boolean
UseWin32	Boolean

Weekdays

Parameter of

ArchiveSchedule on page 199, BandwidthEntry on page 201, BandwidthScheduleEntry on page 206

Properties

Name	Enumeration
None	0
Sunday	1
Monday	2
Tuesday	4
Wednesday	8
Thursday	16
Friday	32
Saturday	64
All	127

Workload

Returned by

Add-DtUvraPhysicalRule on page 20, Get-DtWorkload on page 78

Parameter of

Add-DtUvraPhysicalRule on page 20, Get-DtRecommendedJobOptions on page 58, JobOptions on page 268, New-DtWorkload on page 99

Properties

Name	Type
LogicalRules	String [ ]
PhysicalRules	PhysicalRule [ ] on page 297
WorkloadTypeName	String

WorkloadSupportSummary

Parameter of

WorkloadType on page 375

Properties

Name	Type
Reason	String
ReasonFormatParameters	String

WorkloadType

Returned by

Get-DtWorkloadType on page 80

Properties

Name	Type
IsLicensed	Boolean
IsPresent	Boolean
Name	String
SupportSummary	WorkloadSupportSummary on page 374

Chapter 4 Scripting examples

Below are links to sample Double-Take PowerShell scripts. The sample scripts must be modified. They cannot be used as-is. Modify them to fit your environment. If you need basic assistance with script modifications, contact Technical Support. Assistance with advanced scripting will be referred to Professional Services.

Job creation scripts

Below are links to sample job creation scripts. The sample scripts must be modified. They cannot be used as-is. Modify them to fit your environment. If you need basic assistance with script modifications, contact Technical Support. Assistance with advanced scripting will be referred to Professional Services.

Creating a files and folders job

The following sample script will create a simple files and folders job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtProtectionRules =
- $DtJobOptions =
- $DtJobGuidForFilesAndFolders =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple Double-Take files and folders job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Create a workload
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -WorkloadTypeName "FilesAndFolders"

# Specify the files and folders to protect
$DtProtectionPath = New-Object DoubleTake.Common.Contract.PhysicalRule -Property @{Path="C:\DirName"}
$DtProtectionRules = Add-DtPhysicalRule -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -Rule
$DtProtectionPath
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid

# Get the job options that will be used to create the job
$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtTarget -Source $DtSource -JobType
"FilesAndFolders" -Workload $DtWorkload

# Sets the path mapping on the target to an all-to-one location
$DtJobOptions.JobOptions.CoreConnectionOptions.PathTransformations[0].SourcePath = "C:\"
$DtJobOptions.JobOptions.CoreConnectionOptions.PathTransformations[0].TargetPath = "C:\ReplicatedData\"

# Create the job
$DtJobGuidForFilesAndFolders = New-DtJob -ServiceHost $DtTarget -Source $DtSource -JobType
FilesAndFolders -JobOptions $DtJobOptions.JobOptions

# If you do not want to specify job options and instead use the default options, remove the
PathTransformations lines above
# and use the New-DtFilesAndFoldersJob cmdlet, similar to the following line.
# $DtJobGuidForFilesAndFolders = New-DtFilesAndFoldersJob -ServiceHost $DtTarget -Source $DtSource -Path
"C:\" -JobOptions $DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtTarget -JobId $DtJobGuidForFilesAndFolders

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a

PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating a full server job

The following sample script will create a simple full server job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtProtectionItems =
- $DtJobOptions =
- $DtJobOptions.JobOptions.SystemStateOptions.SourceReservedAddress =
- $DtJobOptions.JobOptions.SystemStateOptions.TargetReservedAddress =
- $DtJobGuidForFullServer =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple Double-Take full server job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Set the reserved IP address on the source and target servers
Set-DtOption $DtSource -Setting @{ReservedAddress="10.10.10.29"}
Set-DtOption $DtTarget -Setting @{ReservedAddress="10.10.10.30"}

# Create a workload
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -WorkloadTypeName "FullServerFailover"

# Add what you want to protect to the workload
$DtLogicalItems = Get-DtLogicalItem -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID
$DtProtectionItems = Set-DtLogicalItemSelection -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -
LogicalPath $DtLogicalItems[0].Path
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID

# Get the default options that will be used to create the job
$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtTarget -Source $DtSource -JobType
"FullServerFailover" -Workload $DtWorkload

# Specify the reserved addresses set earlier to be used in the job options to be used for reverse
$DtJobOptions.JobOptions.SystemStateOptions.SourceReservedAddress = $(Get-DtOption $DtSource
ReservedAddress)["ReservedAddress"]
$DtJobOptions.JobOptions.SystemStateOptions.TargetReservedAddress = $(Get-DtOption $DtTarget
ReservedAddress)["ReservedAddress"]

# Create the job
$DtJobGuidForFullServer = New-DtJob -ServiceHost $DtTarget -Source $DtSource -JobType
"FullServerFailover" -Options $DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtTarget -JobId $DtjobGuidForFullServer

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a

PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating a SQL job

The following sample script will create a simple SQL job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtLogicalItem =
- $DtProtectionItems =
- $DtJobOptions =
- $DtJobGuidForSQL =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple Double-Take SQL job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Create a workload
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -WorkloadTypeName "SQL"

# Add what you want to protect to the workload
$DtLogicalItem = Get-DtLogicalItem -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID | Select-Object -
First 1
$DtProtectionItems = Set-DtLogicalItemSelection -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -
LogicalPath $DtLogicalItem.Path
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID

# Get the job options that will be used to create the job
$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtTarget -Source $DtSource -JobType "SQL" -
Workload $DtWorkload

# Create the job
$DtJobGuidForSQL = New-DtJob -ServiceHost $DtTarget -Source $DtSource -JobType "SQL" -Options
$DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtTarget -JobId $DtJobGuidForSQL

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating an Exchange job

The following sample script will create a simple Exchange job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtExchangeCredentials =
- $DtDomainCredentials =
- $DtLogicalItem =
- $DtProtectionItems =
- $DtJobOptions =
- $DtJobGuidForExchange =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple Double-Take Exchange job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Set Exchange and domain credentials
$DtExchangeCredentials = New-Object DoubleTake.Common.Contract.Credentials $(Get-Credential
domain\administrator)
$DtDomainCredentials = New-Object DoubleTake.Common.Contract.Credentials $(Get-Credential
domain\administrator)

# Create a workload
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -WorkloadTypeName "Exchange"

# Add what you want to protect to the workload
$DtLogicalItem = Get-DtLogicalItem -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID | Select-Object -
First 1
$DtProtectionItems = Set-DtLogicalItemSelection -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -
LogicalPath $DtLogicalItem.Path
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID

# Get the job options that will be used to create the job
$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtTarget -Source $DtSource -JobType "Exchange"
-Workload $DtWorkload

# Specify the Exchange and domain credentials in the job options
$DtJobOptions.JobOptions.ApplicationOptions.ExchangeCredentials = $DtExchangeCredentials
$DtJobOptions.JobOptions.DnsOptions.Domains[0].Credentials = $DtDomainCredentials

# Create the job
$DtJobGuidForExchange = New-DtJob -ServiceHost $DtTarget -Source $DtSource -JobType "Exchange" -
JobOptions $DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtTarget -JobId $DtJobGuidForExchange

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a

PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating a full server to ESX job

The following sample script will create a simple full server to ESX job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtLogicalItem =
- $DtProtectionItems =
- $DtJobOptions =
- $DtJobGuidForEVRA =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple Double-Take full server to ESX job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Create ESX host appliance object
# If you are using vCenter, specify your vCenter. Only specify an ESX host if you are using ESX
standalone.
$VimTarget = New-DtServer -Name gamma -Username root -Password password -Role TargetVimServer
$OtherServers = @($VimTarget)

# Create a workload
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -WorkloadTypeName "VRA"

# Add what you want to protect to the workload
$DtLogicalItem = Get-DtLogicalItem -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID | Select-Object -
First 1
$DtProtectionItems = Set-DtLogicalItemSelection -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -
LogicalPath $DtLogicalItem.Path
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID

# Get the job options that will be used to create the job
$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtTarget -Source $DtSource -OtherServers
$OtherServers -JobType "VRA" -Workload $DtWorkload

# Create the job
$DtJobGuidForEVRA = New-DtJob -ServiceHost $DtTarget -Source $DtSource -OtherServers $OtherServers -
JobType "VRA" -JobOptions $DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtTarget -JobId $DtJobGuidForEVRA

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
Disconnect-DtServer -ServiceHost $VimTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a

PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating a full server to ESX appliance job

The following sample script will create a simple full server to ESX appliance job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtJobOptions =
- $DtJobGuidForUVRA =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple Double-Take full server to ESX appliance job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation.
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and appliance objects
$DtSource = New-DtUvraServer -Name alpha -UserName domain\administrator -Password password -Port 6325
$DtAppliance = New-DtUvraServer -Name beta -UserName root -Password password -Port 6325

# Create ESX host object using default port of 443
# If you are using vCenter, specify your vCenter. Only specify an ESX host if you are using ESX
standalone.
$VimTarget = New-DtUvraServer -Name gamma -Username root -Password password
$OtherServers = @($VimTarget)

# Get proxy and host information
$DtProxy = Get-DtProxiedServerInfo -ServiceHost $DtAppliance -Source $DtSource
$VmHost = Get-DtUvraVmHost -ServiceHost $DtAppliance -EsxHost $VimTarget -VCenter

# Get appliance server info
$DtApplianceServerInfo = Get-DtServerInfo -ServiceHost $DtAppliance

# Get the job options that will be used to create the job
$DtJobOptions = Get-DtUvraRecommendedJobOptions -ServiceHost $DtAppliance -Source $DtSource -SourceInfo
$DtProxy -TargetInfo $DtApplianceServerInfo -ApplianceHost $VmHost

# If you need to modify a job option, you could use a command similar to the following
# $DtJobOptions.JobOptions.VRAOptions.ReplicaVmInfo.DisplayName = "Replica_Name"

# Create the job
$DtJobGuidForUVRA = New-DtJob -ServiceHost $DtAppliance -Source $DtSource -OtherServers $OtherServers -
JobType "UVRA" -JobOptions $DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtAppliance -JobId $DtJobGuidForUVRA

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtAppliance
Disconnect-DtServer -ServiceHost $VimTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating a full server to Hyper-V job

The following sample script will create a simple full server to Hyper-V job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtLogicalItem =
- $DtProtectionItems =
- $DtJobOptions =
- $DtJobGuidForHVRA =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple Double-Take full server to Hyper-V job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Create a workload
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -WorkloadTypeName "VRA"

# Add what you want to protect to the workload
$DtLogicalItem = Get-DtLogicalItem -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID | Select-Object -
First 1
$DtProtectionItems = Set-DtLogicalItemSelection -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -
LogicalPath $DtLogicalItem.Path
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID

# Get the job options that will be used to create the job
$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtTarget -Source $DtSource -JobType "VRA" -
Workload $DtWorkload

# Create the job
$DtJobGuidForHVRA = New-DtJob -ServiceHost $DtTarget -Source $DtSource -JobType "VRA" -JobOptions
$DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtTarget -JobId $DtJobGuidForHVRA

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating an agentless Hyper-V job

The following sample script will create a simple agentless Hyper-V job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $Vm =
- $DtProtectionItems =
- $DtJobOptions =
- $DtJobGuidForDTHV =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple Double-Take agentless Hyper-V job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Create a workload
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -WorkloadTypeName "DTHV"

# Add virtual machine that you want to protect to the workload
$Root = Get-DtLogicalItem -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID
$Vm = Get-DtLogicalItem -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID -RefItem $Root | Where-Object
{ $_.Name -eq "virtual_machine_name" }
$DtProtectionItems = Set-DtLogicalItemSelection -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -
LogicalPath $Vm.Path
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID

# Get the job options that will be used to create the job
$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtTarget -Source $DtSource -JobType "DTHV" -
Workload $DtWorkload

# Create the job
$DtJobGuidForDTHV = New-DtJob -ServiceHost $DtTarget -Source $DtSource -JobType "DTHV" -JobOptions
$DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtTarget -JobId $DtJobGuidForDTHV

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating a data migration job

The following sample script will create a simple data migration job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtProtectionRules =
- $DtJobOptions =
- $DtJobGuidForDataMigration =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple data migration job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
# Create source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Create a workload
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -WorkloadTypeName "MoveDataOnlyMigration"

# Specify the files and folders to protect
$DtProtectionPath = New-Object DoubleTake.Common.Contract.PhysicalRule -Property @{Path="C:\DirName"}
$DtProtectionRules = Add-DtPhysicalRule -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -Rule
$DtProtectionPath
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid

# Get the job options that will be used to create the job
$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtTarget -Source $DtSource -JobType
"MoveDataOnlyMigration" -Workload $DtWorkload

# Sets the path mapping on the target to an all-to-one location
$DtJobOptions.JobOptions.CoreConnectionOptions.PathTransformations[0].SourcePath = "C:\"
$DtJobOptions.JobOptions.CoreConnectionOptions.PathTransformations[0].TargetPath = "C:\ReplicatedData\"

# Create the job
$DtJobGuidForDataMigration = New-DtJob -ServiceHost $DtTarget -Source $DtSource -JobType
MoveDataOnlyMigration -JobOptions $DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtTarget -JobId $DtJobGuidForDataMigration

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating a full server migration job

The following sample script will create a simple full server migration job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtProtectionItems =
- $DtJobOptions =
- $DtJobGuidForFullServerMigration =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple Double-Take full server migration job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Create a workload
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -WorkloadTypeName "MoveServerMigration"

# Add what you want to protect to the workload
$DtLogicalItems = Get-DtLogicalItem -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID
$DtProtectionItems = Set-DtLogicalItemSelection -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -
LogicalPath $DtLogicalItems[0].Path
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID

# Get the default options that will be used to create the job
$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtTarget -Source $DtSource -JobType
"MoveServerMigration" -Workload $DtWorkload

# Create the job
$DtJobGuidForFullServerMigration = New-DtJob -ServiceHost $DtTarget -Source $DtSource -JobType
"MoveServerMigration" -Options $DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtTarget -JobId $DtJobGuidForFullServerMigration

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating a full server to ESX migration job

The following sample script will create a simple full server to ESX migration job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtLogicalItem =
- $DtProtectionItems =
- $DtJobOptions =
- $DtJobGuidForVraMove =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple Double-Take full server to ESX migration job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Create ESX host appliance object
# If you are using vCenter, specify your vCenter. Only specify an ESX host if you are using ESX
standalone.
$VimTarget = New-DtServer -Name gamma -Username root -Password password -Role TargetVimServer
$OtherServers = @($VimTarget)

# Create a workload
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -WorkloadTypeName "VraMove"

# Add what you want to protect to the workload
$DtLogicalItem = Get-DtLogicalItem -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID | Select-Object -
First 1
$DtProtectionItems = Set-DtLogicalItemSelection -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -
LogicalPath $DtLogicalItem.Path
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID

# Get the job options that will be used to create the job
$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtTarget -Source $DtSource -JobType "VraMove" -
Workload $DtWorkload -OtherServers $OtherServers

# Create the job
$DtJobGuidForVraMove = New-DtJob -ServiceHost $DtTarget -Source $DtSource -OtherServers $OtherServers -
JobType "VraMove" -JobOptions $DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtTarget -JobId $DtJobGuidForVraMove

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
Disconnect-DtServer -ServiceHost $VimTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a

PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating a full server to Hyper-V migration job

The following sample script will create a simple full server to Hyper-V migration job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtLogicalItem =
- $DtProtectionItems =
- $DtJobOptions =
- $DtJobGuidForVraMove =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to create a simple Double-Take full server to Hyper-V migration job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Create a workload
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -WorkloadTypeName "VraMove"

# Add what you want to protect to the workload
$DtLogicalItem = Get-DtLogicalItem -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID | Select-Object -
First 1
$DtProtectionItems = Set-DtLogicalItemSelection -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -
LogicalPath $DtLogicalItem.Path
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGUID

# Get the job options that will be used to create the job
$DtJobOptions = Get-DtRecommendedJobOptions -ServiceHost $DtTarget -Source $DtSource -JobType "VraMove" -
Workload $DtWorkload

# Create the job
$DtJobGuidForVraMove = New-DtJob -ServiceHost $DtTarget -Source $DtSource -JobType "VraMove" -JobOptions
$DtJobOptions.JobOptions

# Start the job
Start-DtJob -ServiceHost $DtTarget -JobId $DtJobGuidForVraMove

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Job information scripts

Below are links to sample job information scripts. The sample scripts must be modified. They cannot be used as-is. Modify them to fit your environment. If you need basic assistance with script modifications, contact Technical Support. Assistance with advanced scripting will be referred to Professional Services.

Viewing job information

The following sample script will gather and view job information, including the job ID, the job status, the high and low level options the job is using, and the job statistics. The low-level options gathered and displayed by this script are for a full server job. You will need to modify this script to fit your environment and configuration.

```
# Sample script to gather and view Double-Take job information
# This script is based on an existing full server job

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Login to your source and target servers
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Gather and display the job ID
$DtFSJobId = $(Get-DtJob -ServiceHost $DtTarget).Id
write-output " "
write-output " "
write-output "====================="
write-output "This is the job ID for the job running on the following server: "$DtTarget.HostName""
$DtFSJobId

# Gather and display the job status
$DtFSJobStatus = $(Get-DtJob -ServiceHost $DtTarget).Status
write-output " "
write-output " "
write-output "====================="
write-output "This is the job status."
$DtFSJobStatus

# Gather and display the high-level options being used in the job
$DtFSJobOptions = $(Get-DtJob -ServiceHost $DtTarget).Options
write-output " "
write-output " "
write-output "====================="
write-output "These are the options the job is currently using."
$DtFSJobOptions

# Gather and display the low-level options being used in the job
# These options are specific to full server jobs
$DtFSJobOptionsDetailed = $(Get-DtJob -ServiceHost $DtTarget).Options.FullServerFailoverOptions
write-output " "
write-output " "
write-output "====================="
write-output "These are the specific full server options the job is currently using."
$DtFSJobOptionsDetailed

# Gather and display the statistics for the job
$DtFSJobStatistics = $(Get-DtJob -ServiceHost $DtTarget).Statistics.CoreConnectionDetails
write-output " "
write-output " "
write-output "====================="
write-output "These are the statistics for the job."
$DtFSJobStatistics

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Viewing job Event messages

Most Double-Take Event messages are located in the Application Log with a Source of Double-Take or Double-Take Management Service. You will also find some Event messages in the System log under RepDrv. See the *User's Guide* for details on all of the Double-Take Event messages.

The following sample scripts will gather Double-Take specific Event messages. The cmdlets used in these scripts are not Double-Take cmdlets. They are Windows PowerShell cmdlets. See your Windows PowerShell documentation for more details and examples on how to use these cmdlets.

You will need to modify this script to fit your environment and configuration.

Each of the Get-EventLog lines in the sample scripts are wrapped to the line below so that you can see all of the text on the page. When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to view job Event messages

# Set the date for how far back you want to view
$Date = get-date 04/29/2013

# Display all Double-Take and Double-Take Management Service Event messages
# since the date you specified
Get-EventLog -LogName Application -Source @("Double-Take", "Double-Take Management Service") -After $Date
```

```
# Sample script to view job Event messages

# Display the last five Double-Take or Double-Take Management Service Event
# messages, listing all properties of the Events
Get-EventLog -LogName Application -Source @("Double-Take", "Double-Take Management Service") -Newest 5 |
format-list -property *
```

```
# Sample script to view job Event messages

# Set the values of the Event IDs you want to see
$FirstEventId = 4065     # Target data state change
$SecondEventId = 4111    # Sharing violation on target
$ThirdEventId = 8196     # Memory issues on source

# Set the date for how far back you want to view
$Date = get-date 04/29/2013

# Display specific Double-Take or Double-Take Management Service Event messages
# based on the Event IDs
Get-EventLog -LogName Application -Source @("Double-Take", "Double-Take Management Service") -After $Date
| Where-Object {$_.EventID -eq $FirstEventId -or $_.EventId -eq $SecondEventId}

# Display specific RepDrv Event messages based on the Event IDs
Get-EventLog -LogName System -Source RepDrv -After $Date | Where-Object {$_.EventID -eq $ThirdEventId}
```

```
# Sample script to view job Event messages

# Display specific Double-Take or Double-Take Management Service Event messages
# based on the Event index number, and listing all properties of the Event
Get-EventLog -LogName Application -Source @("Double-Take", "Double-Take Management Service") | Where-
Object {$_.Index -eq 99461} | format-list -property *
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Creating a job diagnostics file

The following sample script will create a job diagnostics file, also known as DTInfo. The file will be located in the \Service\Data directory where you installed Double-Take. This is a file you may want to give to technical support if you are troubleshooting a job. There will be a separate file for each job on your target. You will need to modify this script to fit your environment and configuration.

```
# Sample script to create a job diagnostics files

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Login to your target server
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Get the jobs on the target and pass through to create a diagnostics file
Get-DtJob -ServiceHost $DtTarget | Save-DtJobDiagnostics -ServiceHost $DtTarget

# Close the connections for the server object
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Job control scripts

Below are links to sample job control scripts. The sample scripts must be modified. They cannot be used as-is. Modify them to fit your environment. If you need basic assistance with script modifications, contact Technical Support. Assistance with advanced scripting will be referred to Professional Services.

Validating an existing job

The following sample script will validate an existing job. You will need to modify this script to fit your environment and configuration.

The $DtJob line is wrapped to the line below so that you can see all of the text on the page. When re-creating a script like this for your environment, make sure you enter that command on just one line.

```
# Sample script to validate an existing Double-Take job

# Create variables for the source and target server names
$DtSourceName="alpha"
$DtTargetName="beta"

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create a server object for the target.
$DtTarget = New-DtServer -Name $DtTargetName -UserName domain\administrator -Password password

# Find the appropriate job, based on the source server name.
$DtJob = Get-DtJob -ServiceHost $DtTarget | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DTSourceName}

# Validate the job options.
$DtConfirmation = Confirm-DtJobOptions -ServiceHost $DtTarget -JobId $DtJob.Id -JobOptions $DtJob.Options

# Give the validation process time to complete.
while ($true)
{
          sleep 1
          $DtConfirmStatus = Get-DtVerificationStatus -ServiceHost $DtTarget -Token $DtConfirmation
          if ($DtConfirmStatus.Task.Status -eq "Faulted")
          {
                    throw $("Validation failed: {0}" -f $DtConfirmStatus.Task.MessageId)
          }
          if ($DtConfirmStatus.Task.Status -eq "Completed")
          {
                    break
          }
}
$StatusCount=0
$DtConfirmStatus.Steps | ForEach-Object {
          if ($_.Status -eq "Warning" -or $_.Status -eq "Error")
          {
                    $StatusCount++
                    # For each error or warning, display the level and message.
                     Write-Host "$($_.Status) : $($_.MessageKey)"
          }
}

# Identify if there were no errors or warnings.
if ($StatusCount -eq 0)
{
          Write-Host "No job validation errors or warnings were detected."
}

# Close the connection for the server object
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a

PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Editing a files and folders job

The following sample script will edit an existing files and folders job. You will need to modify this script to fit your environment and configuration.

The following lines are wrapped to the line below so that you can see all of the text on the page.

- $DtJob =
- $DtNewRule =
- $DtExcludeTxtRule =

When re-creating a script like this for your environment, make sure you enter those commands on just one line.

```
# Sample script to edit an existing files and folders Double-Take job

# Create variables for the source and target server names
$SourceName="alpha"
$TargetName="beta"

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name $SourceName -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name $TargetName -UserName domain\administrator -Password password

# Identify the job, based on the source server name
$DtJob = Get-DtJob -ServiceHost $DtTarget | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $SourceName}

# Create a workload object on the source to edit the current workload rules
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -Workload $DtJob.Options.Workload

# Specify the additional files and folders to protect
$DtNewRule = Add-DtPhysicalRule -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -Path
"C:\NewDirectory" -Recurse

# Specify files to exclude from protection, in this example .txt files in the new protection rule
$DtExcludeTxtRule = Add-DtPhysicalRule -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid -Path
"C:\NewDirectory\*.txt" -Recurse -Exclude

# Update the workload rules in the job options with the new modifications
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid
$DtJob.Options.Workload=$DtWorkload

# Update the path mapping of the replicated data on the target based on the current recommendations
$DtTargetPath = Get-DtRecommendedPathTransform -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid
$DtJob.Options.CoreConnectionOptions.PathTransformations = $DtTargetPath

# If you do not want to use the one-to-one path mapping in the default recommended options,
# you can configure the job to use specific locations, similar to the following lines.
# $DtJob.Options.CoreConnectionOptions.PathTransformations[0].SourcePath = "C:\"
# $DtJob.Options.CoreConnectionOptions.PathTransformations[0].TargetPath = "C:\ReplicatedData\"

# Verify the new job options on the existing job.
$DtConfirmation = Confirm-DtJobOptions -ServiceHost $DtTarget -JobId $DtJob.Id -JobOptions $DtJob.Options
do
{
          # Poll every second for the confirmation status
          Start-Sleep -Seconds 1
          $DtConfirmStatus = Get-DtVerificationStatus -ServiceHost $DtTarget -Token $DtConfirmation
          # When the ActivityCompletionStatus is not InProgress, the confirmation is complete.
}
while ($DtConfirmStatus.VerificationStatus.Step.Status -eq 0)
```

```
# If the ActivityCompletionStatus is Error, print out the steps reporting an Error.
if ($DtConfirmStatus.VerificationStatus.Step.Status -eq 3)
{
          Write-Error "The following job validation errors were detected:"
          $DtConfirmStatus.Steps | ForEach-Object
          {
                    if ($_.Status -eq 3)
                    {
                              Write-Error "$($_.Id) : $($_.TitleKey) : $($_.MessageKey)"
                    }
          }
          # Terminate so the job is not edited with invalid options
          throw "Job validation failure."
}

# Apply new job options with the updated workload rules, forcing a remirror.
Edit-DtJob -ServiceHost $DtTarget -JobId $DtJob.Id -JobOptions $DtJob.Options

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Changing the compression setting for an existing job

The following sample script will change the compression setting for an existing job. You will need to modify this script to fit your environment and configuration.

The $DtJob line is wrapped to the line below so that you can see all of the text on the page. When re-creating a script like this for your environment, make sure you enter that command on just one line.

```
# Sample script to change the compression settings for an existing Double-Take job

# Create variables for the source and target server names
$SourceName="alpha"
$TargetName="beta"

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create source and target objects
$DtSource = New-DtServer -Name $SourceName -UserName administrator -Password password
$DtTarget = New-DtServer -Name $TargetName -UserName administrator -Password password

# Identify the job, based on the source server name
$DtJob = Get-DtJob -ServiceHost $DtTarget | Where-Object {
$_.Statistics.CoreConnectionDetails.SourceMachineName -eq $SourceName}

# Create a workload object on the source to edit the current workload rules
$DtWorkloadGUID = New-DtWorkload -ServiceHost $DtSource -Workload $DtJob.Options.Workload

# Enable compression using one of the following combinations
# level = -1 Compression is disabled
# level = 0 and algorithm = 10 Compression is enabled at low level
# level = 1 and algorithm = 21 Compression is enabled at medium level
# level = 2 and algorithm = 31 Compression is enabled at high level
$DtJob.Options.CoreConnectionOptions.ConnectionStartParameters.CompressionLevel.Level=1
$DtJob.Options.CoreConnectionOptions.ConnectionStartParameters.CompressionLevel.Algorithm=21

# Update the workload rules in the job options with the new modifications
$DtWorkload = Get-DtWorkload -ServiceHost $DtSource -WorkloadId $DtWorkloadGuid
$DtJob.Options.Workload=$DtWorkload

# Verify the new job options on the existing job.
$DtConfirmation = Confirm-DtJobOptions -ServiceHost $DtTarget -JobId $DtJob.Id -JobOptions $DtJob.Options
do
{
          # Poll every second for the confirmation status
          Start-Sleep -Seconds 1
          $DtConfirmStatus = Get-DtVerificationStatus -ServiceHost $DtTarget -Token $DtConfirmation
          # When the activity completion status is not InProgress, the confirmation is complete.
} while ($DtConfirmStatus.VerificationStatus.Step.Status -eq 0)

# If the completion status is Error, print out the steps reporting an Error
if ($DtConfirmStatus.VerificationStatus.Step.Status -eq 3)
{
          Write-Error "The following job validation errors were detected:"
          $DtConfirmStatus.Steps | ForEach-Object
          {
                    if ($_.Status -eq 3)
                    {
                              Write-Error "$($_.Id) : $($_.TitleKey) : $($_.MessageKey)"
                    }
           }
          # Terminate so the job is not edited with invalid options
          throw "Job validation failure."
}

# Apply new job options with the updated workload rules, forcing a remirror.
Edit-DtJob -ServiceHost $DtTarget -JobId $DtJob.Id -JobOptions $DtJob.Options
```

```
# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Stopping and starting a job

The following sample scripts stop and start a Double-Take job on your target. You will need to modify these scripts to fit your environment and configuration.

```
# Sample script to stop a Double-Take job on your target

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create target object
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Get the job ID of the job running on the target
$DtJobId = Get-DtJob -ServiceHost $DtTarget

# Stop the job running on the target
Stop-DtJob -ServiceHost $DtTarget -JobId $DtJobId.Id

# Close the connections for the server object
Disconnect-DtServer -ServiceHost $DtTarget
```

```
# Sample script to start a Double-Take job on your target

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create target object
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Get the job ID of the job running on the target
$DtJobId = Get-DtJob -ServiceHost $DtTarget

# Resume the job running on the target
Start-DtJob -ServiceHost $DtTarget -JobId $DtJobId.Id

# Close the connections for the server object
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

If you have multiple jobs on your target, you can use the Windows Where-Object cmdlet to identify a specific job by its source URI, source server name, or by job name. For example, you might use one of the following.

```
$DtJobId = $(Get-DtJob -ServiceHost $DtTarget | Where-Object { $_.SourceHostUri.Host -eq "ServerName" }).Id

$DtJobId = $(Get-DtJob -ServiceHost $DtTarget | Where-Object { $_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}).Id

$DtJobId = $(Get-DtJob -ServiceHost $DtTarget | Where-Object { $_.Options.Name -eq "source to target" }).Id
```

See your Windows PowerShell documentation for more details on using the Where-Object command.

Pausing and resuming a job

The following sample scripts pause and resume a Double-Take job on your target. You will need to modify these scripts to fit your environment and configuration.

```
# Sample script to pause a Double-Take job on your target

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create target object
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Get the job ID of the job running on the target
$DtJobId = Get-DtJob -ServiceHost $DtTarget

# Pause the job running on the target
Suspend-DtJob -ServiceHost $DtTarget -JobId $DtJobId.Id

# Close the connections for the server object
Disconnect-DtServer -ServiceHost $DtTarget
```

```
# Sample script to resume a Double-Take job on your target

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create target object
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Get the job ID of the job running on the target
$DtJobId = Get-DtJob -ServiceHost $DtTarget

# Resume the job running on the target
Resume-DtJob -ServiceHost $DtTarget -JobId $DtJobId.Id

# Close the connections for the server object
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

If you have multiple jobs on your target, you can use the Windows Where-Object cmdlet to identify a specific job by its source URI, source server name, or job name. For example, you might use one of the following.

```
$DtJobId = $(Get-DtJob -ServiceHost $DtTarget | Where-Object { $_.SourceHostUri.Host -eq "ServerName" }).Id

$DtJobId = $(Get-DtJob -ServiceHost $DtTarget | Where-Object { $_.Statistics.CoreConnectionDetails.SourceMachineName -eq $DtServerObjectAlpha}).Id

$DtJobId = $(Get-DtJob -ServiceHost $DtTarget | Where-Object { $_.Options.Name -eq "source to target" }).Id
```

See your Windows PowerShell documentation for more details on using the Where-Object command.

Viewing and setting job and server options

The following sample script will gather and set several Double-Take job and server options. You may want to consider running cmdlets like this from the PowerShell command line, rather than a script, so you can see the values returned from the get cmdlets and then make appropriate adjustments for your set cmdlets. You will need to modify this script to fit your environment and configuration.

```
# Sample script to gather and set Double-Take job and server options
# You may want to run these cmdlets from the PowerShell command prompt
# so that you can see the values returned for each of the get cmdlets
# and then determine appropriate desired values for each option

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Login to a server by creating source and target objects
$DtSource = New-DtServer -Name alpha -UserName domain\administrator -Password password
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Gather and display several job and server settings
$DtMaxChecksumBlocksSource = Get-DtOption -ServiceHost $DtSource -Name MaxChecksumBlocks
$DtMaxChecksumBlocksTarget = Get-DtOption -ServiceHost $DtTarget -Name MaxChecksumBlocks
$DtMirrorChunkSizeSource = Get-DtOption -ServiceHost $DtSource -Name MirrorChunkSize
$DtMirrorChunkSizeTarget = Get-DtOption -ServiceHost $DtTarget -Name MirrorChunkSize
$DtCalculateByVolumeSource = Get-DtOption -ServiceHost $DtSource -Name CalculateByVolume
$DtCalculateByVolumeTarget = Get-DtOption -ServiceHost $DtTarget -Name CalculateByVolume
$DtAutoRemirrorSource = Get-DtOption -ServiceHost $DtSource -Name AutoRemirror
$DtAutoRemirrorTarget = Get-DtOption -ServiceHost $DtTarget -Name AutoRemirror
write-output "=================="
write-output "These are the current options and values."
write-output "The source is displayed first, and the target is displayed second."
$DtMaxChecksumBlocksSource
$DtMaxChecksumBlocksTarget
$DtMirrorChunkSizeSource
$DtMirrorChunkSizeTarget
$DtCalculateByVolumeSource
$DtCalculateByVolumeTarget
$DtAutoRemirrorSource
$DtAutoRemirrorTarget

# Store the desired value for each job and server setting
$DtMaxChecksumBlocksDesiredValue = @{MaxChecksumBlocks=64}
$DtMirrorChunkSizeDesiredValue = @{MirrorChunkSize=131072}
$DtCalculateByVolumeDesiredValue = @{CalculateByVolume=1}
$DtAutoRemirrorDesiredValue = @{AutoRemirror=1}

# Set the new values
Set-DtOption -ServiceHost $DtSource -Setting $DtMaxChecksumBlocksDesiredValue
Set-DtOption -ServiceHost $DtTarget -Setting $DtMaxChecksumBlocksDesiredValue
Set-DtOption -ServiceHost $DtSource -Setting $DtMirrorChunkSizeDesiredValue
Set-DtOption -ServiceHost $DtTarget -Setting $DtMirrorChunkSizeDesiredValue
Set-DtOption -ServiceHost $DtSource -Setting $DtCalculateByVolumeDesiredValue
Set-DtOption -ServiceHost $DtTarget -Setting $DtCalculateByVolumeDesiredValue
Set-DtOption -ServiceHost $DtSource -Setting $DtAutoRemirrorDesiredValue
Set-DtOption -ServiceHost $DtTarget -Setting $DtAutoRemirrorDesiredValue

# Regather and display the updated values
$DtMaxChecksumBlocksSource = Get-DtOption -ServiceHost $DtSource -Name MaxChecksumBlocks
$DtMaxChecksumBlocksTarget = Get-DtOption -ServiceHost $DtTarget -Name MaxChecksumBlocks
$DtMirrorChunkSizeSource = Get-DtOption -ServiceHost $DtSource -Name MirrorChunkSize
$DtMirrorChunkSizeTarget = Get-DtOption -ServiceHost $DtTarget -Name MirrorChunkSize
$DtCalculateByVolumeSource = Get-DtOption -ServiceHost $DtSource -Name CalculateByVolume
$DtCalculateByVolumeTarget = Get-DtOption -ServiceHost $DtTarget -Name CalculateByVolume
$DtAutoRemirrorSource = Get-DtOption -ServiceHost $DtSource -Name AutoRemirror
$DtAutoRemirrorTarget = Get-DtOption -ServiceHost $DtTarget -Name AutoRemirror
write-output " "
write-output "=================="
write-output "These are the updated options and values."
write-output "The source is displayed first, and the target is displayed second."
$DtMaxChecksumBlocksSource
$DtMaxChecksumBlocksTarget
$DtMirrorChunkSizeSource
```

```
$DtMirrorChunkSizeTarget
$DtCalculateByVolumeSource
$DtCalculateByVolumeTarget
$DtAutoRemirrorSource
$DtAutoRemirrorTarget

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Other sample scripts

Below are links to other sample scripts. The sample scripts must be modified. They cannot be used as-is. Modify them to fit your environment. If you need basic assistance with script modifications, contact Technical Support. Assistance with advanced scripting will be referred to Professional Services.

Pausing and resuming your target

The following sample scripts pause and resume your Double-Take target. (The server itself is not paused. Only Double-Take processing is paused.) You will need to modify these scripts to fit your environment and configuration.

```
# Sample script to pause the Double-Take target

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create target object
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Pause all of the Double-Take jobs on the target
Suspend-DtTarget -ServiceHost $DtTarget -All

# Close the connections for the server object
Disconnect-DtServer -ServiceHost $DtTarget
```

```
# Sample script to resume the Double-Take target

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Create target object
$DtTarget = New-DtServer -Name beta -UserName domain\administrator -Password password

# Resume all of the Double-Take jobs on the target
Resume-DtTarget -ServiceHost $DtTarget -All

# Close the connections for the server object
Disconnect-DtServer -ServiceHost $DtTarget
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Shutting down the Double-Take service on a server

The following sample script will login to a Double-Take server and then shutdown the Double-Take service on that server. You will need to modify this script to fit your environment and configuration.

```
# Sample script to shutdown the Double-Take service on a server

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Login to a server
$DtServer = New-DtServer -Name alpha -UserName domain\administrator -Password password

# Shutdown the Double-Take service on the server
Stop-DtReplicationService -ServiceHost $DtServer

# Close the connections for the server object
Disconnect-DtServer -ServiceHost $DtServer
```

If you want to hide your user credentials in your script, use the Windows PowerShell Get-Credential cmdlet. The password will not be visible because Windows stores an encrypted password. See Hiding your password in a PowerShell script on page 414 for basic details on using this cmdlet. See your Windows PowerShell documentation for detailed instructions.

Hiding your password in a PowerShell script

The Double-Take PowerShell cmdlets require a server object, and that server object requires user credentials for the specified server. Many corporate security policies do not allow for user passwords to be typed in plain text, which can make scripting difficult. You can use the credential object returned from the Windows PowerShell Get-Credential cmdlet. This password will not be visible because Windows stores an encrypted password. The following sample script logs in to a Double-Take server using a hidden password. See your Windows PowerShell documentation for more details on creating a credential object with Get-Credential. You will need to modify this script to fit your environment and configuration.

```
# Sample script to login to a Double-Take server using a hidden password

# Import the Double-Take PowerShell module
# This may be \Service\ or \Console\ depending on your installation
import-module "C:\Program Files\Vision Solutions\Double-Take\Console\DoubleTake.PowerShell.dll"

# Store user credentials in an encrypted form
$DtCredentialEncrypted = Get-Credential domain\administrator

# At this point, you will be prompted to supply the password
# and the credentials will be stored in an encrypted format

# Create source and target objects
$DtSource = New-DtServer -Name alpha -Credential $DtCredentialEncrypted
$DtTarget = New-DtServer -Name beta -Credential $DtCredentialEncrypted

# Close the connections for the server objects
Disconnect-DtServer -ServiceHost $DtSource
Disconnect-DtServer -ServiceHost $DtTarget
```

Chapter 5 DTCL to PowerShell mapping

If you have Double-Take scripts created using the Double-Take Command Language (DTCL), you will need to update your scripts to the new PowerShell cmdlets introduced in Double-Take version 6.0. The following table will help you map the DTCL commands to the new PowerShell cmdlets.

DTCL command	PowerShell cmdlet
Block All	Set-DtPathBlocking
Compression Disable	
Compression Enable	
Compression List	
Compression Set	
ConID	Get-DtConnectionIds
Connect	New-DtFilesAndFoldersJob New-DtJob Start-DtJob
Connect TDU	
Disconnect	Remove-DtJob Stop-DtJob
Email Add	Set-DtEmailNotificationOptions
Email Disable	
Email Enable	Set-DtEmailNotificationOptions
Email Filter	
Email From Address	Set-DtEmailNotificationOptions
Email Get Email Config	Get-DtEmailNotificationOptions
Email Mail Server	Set-DtEmailNotificationOptions
Email Remove	Set-DtEmailNotificationOptions
Email Set Filter Include	Set-DtEmailNotificationOptions
Email Subject	Set-DtEmailNotificationOptions
Email Test	Test-DtEmailNotification
Environment	

DTCL command	PowerShell cmdlet
Exit	
Failback	Start-DtJobFailback
Failover	Start-DtJobFailover
Get	Get-DtActivationStatus Get-DtOption Get-DtPathBlocking Get-DtProductInfo
GetEnvStr	
Get Local	
Help	Get-Help <cmdlet-name>
Limit Bandwidth	Set-DtBandwidthLimit
Limit Bandwidth Disable	
Limit Bandwidth Schedule Add	Set-DtBandwidthLimit
Limit Bandwidth Schedule Clear	
Limit Bandwidth Schedule Enable	Set-DtBandwidthLimit
Limit Bandwidth Schedule List	Get-DtBandwidthLimit
Limit Bandwidth Schedule Remove	
Load Source	
Load System State	
Load Target	
Login	
Logout	
Mirror Pause	Suspend-DtMirror
Mirror Resume	Resume-DtMirror
Mirror Start	Start-DtMirror
Mirror Stop	Stop-DtMirror
Monitor Account Add	

DTCL command	PowerShell cmdlet
Monitor Account Remove	
Monitor Clear	
Monitor Create	New-DtFilesAndFoldersJob New-DtJob
Monitor Delete	Remove-DtJob
Monitor Display	Get-DtRecommendedJobOptions
Monitor Get	
Monitor List	
Monitor Move	New-DtFilesAndFoldersJob New-DtJob
Monitor Option	New-DtFilesAndFoldersJob New-DtJob
Monitor Remove	
Monitor Script Add	New-DtFilesAndFoldersJob New-DtJob
Monitor Script Remove	
Monitor Start	Start-DtJob
Monitor Stop	Stop-DtJob
Monitor Use	
NIC List	
Orphans Disable	
Orphans Enable	
Orphans Start	Start-DtOrphansProcessing
Orphans Stop	
Pause Target	Suspend-DtTarget
Ping	
Queue Task	New-DtTaskParameters Invoke-DtQueueTask

DTCL command	PowerShell cmdlet
Quit	
Replication Start	Start-DtReplication
Replication Stop	Stop-DtReplication
Repset Calculate	
Repset Create	New-DtWorkload
Repset Delete	
Repset Display	Get-DtJob
Repset List	Get-DtWorkloadPhysicalItem
Repset Resync	
Repset Rule Add	Add-DtPhysicalRule
Repset Rule Remove	Remove-DtPhysicalRule
Repset Save	
Repset Use	
Restore	Start-DtJobRestore
Resume Target	Resume-DtTarget
Schedule Clear	
Schedule Disable	
Schedule Enable	
Schedule End	
Schedule Start	
Schedule Window	
Script Account	Set-DtScriptCredentials
Script Add	
Script List	Get-DtJob
Script Remove	
Script Test	Test-DtScript
Set	Set-DtActivationCode Set-DtOption

DTCL command	PowerShell cmdlet
Set Local	
Shutdown	Restart-DtReplicationService Stop-DtReplicationService
Snapshot Create	Checkpoint-DtConnection
Snapshot Delete	Remove-DtSnapshot
Snapshot List	Get-DtSnapshot
Snapshot Revert	
Snapshot Schedule Disable	
Snapshot Schedule Display	Get-DtJob
Snapshot Schedule Enable	
Snapshot Schedule Every	
Source	New-DtServer
StatsLog Start	
StatsLog Status	
StatsLog Stop	
Status	
Target	New-DtServer
Test Connections	
Time Now	
Transmission Pause	Suspend-DtTransmission
Transmission Resume	Resume-DtTransmission
Transmission Start	Start-DtTransmission
Transmission Stop	
Unblock All	Set-DtPathBlocking
Unload Source	
Unload System State	
Unload Target	

DTCL command	PowerShell cmdlet
Verify	Start-DtVerify
Version	Get-DtProductInfo
Wait	
Wait on Mirror	Wait-DtMirrorComplete
Wait on Restore	
Wait on Target	
Write	

Chapter 6 Server and job settings

The easiest way to view and change select server and job settings is through the Double-Take Console. However, not all of the settings are available there. To view and update the remaining settings, in addition to the settings available in the console, you can use the cmdlets Get-DtOption on page 47 and Set-DtOption on page 129 to view and modify the settings. .

The following table lists all of the settings, in decimal value.

Double-Take Availability and Double-Take Move share the same set of server and job settings. Some settings apply to one product, some to the other, and some to both. For settings that apply to both, the Double-Take Availability terminology is used. For example, PreFailoverScript is used for the script to be run before failover or cutover.

AcquireDataRetryLimit

Description—The length of time, in milliseconds, spent retrying a file read if there is a read error

Values—Any positive, integer value

Default—2000

Console Setting—None

Service restart required—No

ActivationCode

Description—24-character Double-Take activation code

Values—Unique value for each customer

Default—N/A

Console Setting—Edit Server Properties page, Licensing section, Current activation codes

Service restart required—No

AdapterFlags

Description—Specifies the adapter to use when establishing a connection. This option should not be changed.

Values—2 Encryption, 4 Network Data Representation

Default—4

Console Setting—None

Service restart required—Yes

AddOnCodes

Description—This setting is no longer used.

Advertisement

Description—Indicates if the server uses Active Directory to advertise itself so that the Double-Take Console can be populated through automatic discovery

Values—0 Do not use Active Directory advertisement, 8 Use Active Directory advertisement

Default—8

Console Setting—Edit Server Properties page, Setup section, Advertise service with Active Directory

Service restart required—Yes

Notes—If Active Directory advertisement is enabled, there is a 200 byte impact on the Active Directory service for each server that registers. The Double-Take service registers with Active Directory at startup and unregisters at shutdown.

AllFailover

Description—Specifies which IP addresses to failover

Values—0 Failover only monitored IP addresses, 1 Failover all IP addresses

Default—1

Console Setting—Set Options page, Failover Options section, Failover IP addresses

Service restart required—No

AllMustFail

Description—Specifies whether or not all IP addresses must fail for failover to take place

Values—0 any IP address can fail, 1 All IP addresses must fail

Default—1

Console Setting—None

Service restart required—No

ArchiveExclusionDirectories

Description—This setting is no longer used.

ArchiveExclusionFiles

Description—This setting is no longer used.

ArchiveLoopAttempts

Description—This setting is no longer used.

ArchiveLoopDelay

Description—This setting is no longer used.

ArchivePreviewFileName

Description—This setting is no longer used.

ArchivePreviewMaxFileSize

Description—This setting is no longer used.

ArchiveRemoveBinFileOnRecall

Description—This setting is no longer used.

ArchiveRequireMirrorCompleteTime

Description—This setting is no longer used.

ArchiveUseAlternateDate

Description—This setting is no longer used.

ArchiveUseDNSName

Description—This setting is no longer used.

AutoCalcEulaAccepted

Description—Used internally by Double-Take. Do not modify this entry.

AutoReconnect

Description—Specifies whether to reinstate the target connection(s) when the source machine is brought online after a source machine failure

Values—0 Do not reconnect, 1 Reconnect

Default—1

Console Setting—Edit Server Properties page, Setup section, Automatically reconnect during source initialization

Service restart required—Yes

AutoRemirror

Description—Specifies whether to remirror when a source is brought online after an auto-disconnect

Values—0 Do not remirror, 1 Perform a file differences checksum mirror, 2 Perform a full mirror, 3 Perform a file differences mirror, 4 Perform a date comparison mirror and send data only if the source data is newer than the target data.

Default—1

Console Setting—Edit Server Properties page, Setup section, Behavior when automatically reconnecting

Service restart required—No

AutoRemirrorRetry

Description—Specifies how often, in seconds, the source should check for connections that have been reconnected but still need to be remirrored

Values—any integer

Default—30

Console Setting—None

Service restart required—No

AutoRetransmit

Description—Determines whether or not a source that has lost its connection with a target will attempt to reconnect to the target

Values—0 Do not attempt to reconnect, 1 Attempt to reconnect

Default—1

Console Setting—None

Service restart required—No

BackupDir

Description—Location on the target of the backup of the protected data sets

Values—any valid path

Default—the location where the Double-Take files were installed

Console Setting—None

Service restart required—No

CalculateByVolume

Description—Calculates the approximate size of a protected data set by using the size of the volume and subtracting the free space

Values—0 Disabled, 1 Enabled

Default—0

Console Setting—None

Service restart required—No

Notes—If your protected data set contains a large number of files, for example, 250,000 or more, you may want to disable the calculation of the protected data set size so that data will start being mirrored sooner. If calculation is enabled, the source calculates the file size before it starts mirroring. This can take a significant amount of time depending on the number of files and system performance. Disabling calculation will result in the mirror status not showing the percentage complete or the number of bytes remaining to be mirrored. CalculateByVolume can be enabled as a workaround. This setting will get the amount of disk space in use for the entire volume from the operating system, so the calculation occurs instantaneously. However, if the entire volume is not being replicated, the mirror percentage complete and bytes remaining will be incorrect accordingly.

Do not use enable this option if you are using a full server job because it will bypass needed hard link processing.

CalculateOnConnect

Description—Specifies whether or not the amount of data to be mirrored should be calculated on connection

Values—0 Do not calculate on connection, 1 Calculate on connection

Default—1

Console Setting—None

Service restart required—No

CaseSensitiveRepSetQueries

Description—This entry is no longer used.

ChangeJournalState

Description—An internal setting for change journal tracking. Do not modify this setting.

ChangeJournalSystemState

Description—An internal setting for change journal tracking. Do not modify this setting.

ChecksumAll

Description—Setting to allow for the difference checksum option on mirror, verify, or restore to ignore the date, time, and size of the file and perform a checksum calculation on all files

Values—0 Checksum using date, time, size comparison, 1 Checksum all files regardless of the date, time, or file size

Default—1

Console Setting—Edit Server Properties page, Source section, Use block checksum during difference mirrors

Service restart required—No

ClusterDir

Description—Location of a Microsoft Cluster Service installation, if it exists

Values—any valid path

Default—determined by the Microsoft Cluster Service installation

Console Setting—None

Service restart required—No

ConnectionFile

Description—Name of the database file containing connection information

Values—any valid file name

Default—connect.sts

Console Setting—None

Service restart required—No

CreateDumpOnAckErrors

Description—Enables additional logging for out of order acknowledgement errors

Values—0 Do not create a logging file, 1 Create a logging file

Default—0

Console Setting—None

Service restart required—No

DataPath

Description—The location of the Double-Take file attribute, protected data set, connection, and schedule database files

Values—any valid path

Default—the location where the Double-Take files were installed

Console Setting—None

Service restart required—No

DefaultAddress

Description—The default primary IP address in a multi-homed server

Values—any valid IP address that will act as your primary IP address for connecting the source to the target

Default—<null>

Console Setting—Edit Server Properties page, General section, Default address

Service restart required—Yes

DefaultProtocol

Description—The default protocol

Values—2 IPv4 protocol only, 23 IPv4 and IPv6 protocols, 3 TDU (Throughput Diagnostics Utility)

Default—2 for Windows 2003, 23 for Windows 2008 and 2012

Console Setting—None

Service restart required—Yes

DefaultReaderType

Description—Internal setting used by Double-Take RecoverNow for recoveries. Do not modify this setting.

DelayGCArbitration

Description—Number of seconds to delay the arbitration process. This option allows time for the network to become stable before trying to execute arbitration logic, for example, when a cluster failover has occurred, but the network has a lag before it returns to a stable state. Arbitration should not start until the network is back in that stable state.

Values—any positive number

Default—0

Console Setting—None

Service restart required—No

DelayGCConnection

Description—Delays the GeoCluster Replicated Disk resource connection to allow the cluster service enough time to reset

Values—1-15

Default—3

Console Setting—None

Service restart required—No

DiffMirrorHardLinkCleanup

Description—Specifies if files with more than one hard link are deleted on the target during a difference mirror and then relinked after the remirror is complete. This setting only applies to Windows 2008 and 2012 servers with a full server job or full server migration job. If mirror performance is negatively impacted by this setting, you may want to disable it.

Values—0 Hard link files are not deleted and relinked during a difference mirror, 1 Hard link files are deleted and relinked during a difference mirror

Default—1

Console Setting—None

Service restart required—No

DirUNetPort

Description—Port used by pre-5.2 versions for directed UDP communications

Values—1025 - 65535

Default—1105

Console Setting—None

Service restart required—Yes

DisableAttributeReplication

Description—Specifies whether or not attributes (read-only, hidden, and so on) are replicated to the target

Values—0 Enable attribute replication, 1 Disable attribute replication

Default—0

Console Setting—None

Service restart required—No

DropOpOnAccessDeniedError

Description—Specifies whether or not operations are dropped or retried after an access denied error

Values—0 The operation will be retried, 1 The operation will be dropped

Default—1

Console Setting—None

Service restart required—No

DropOpOnHandleError

Description—Determines if an additional attempt is made to access a file by a Microsoft API call if the Double-Take call fails.

Values—0 When opening a file using the Double-Take driver fails, attempt to open the file usingthe Microsoft Win32 API, 1 When opening a file using the Double-Take driver fails, skip the file and document it in the Double-Take log

Default—1

Console Setting—None

Service restart required—No

Notes—If the value is set to 0 and the Win32 call also fails, Double-Take will skip the file and document it in the Double-Take log

DTSetupType

Description—Used by the Double-Take installation program to maintain the installation settings for an upgrade. Do not modify this setting.

DumpDiskQuotaIntervalMinutes

Description—Specifies how often, in minutes, a snapshot of the disk quota is taken as a backup in case the live registry is not usable at failover or cutover

Values—any integer

Default—240

Console Setting—None

Service restart required—No

DumpHiveIntervalMinutes

Description—Specifies how often, in minutes, a snapshot of the registry is taken as a backup in case the live registry is not usable at failover or cutover

Values—any integer

Default—240

Console Setting—None

Service restart required—No

EmailEnabled

Description—Specifies if e-mail notification is enabled

Values—0 E-mail notification is disabled, 1 E-mail notification is enabled

Default—0

Console Setting—Edit Server Properties page, E-mail Notification section, Enable e-mail notifications

Service restart required—Yes from the registry, No from the console

Notes—This is a read-only setting. If you change this setting using the registry editor, e-mail notification will not automatically start. You must use the console or a PowerShell script to start e-mail notification.

EmailExcludeIds

Description—Identifies the Windows Event Viewer messages that are excluded from e-mail notification.

Values—Comma or semicolon separated list of Event Viewer IDs. You can indicate ranges within the list.

Default—None

Console Setting—Edit Server Properties page, E-mail Notification section, Exclude these event IDs

Service restart required—Yes from the registry, No from the console

EmailFromAddress

Description—Specifies the e-mail address that will appear in the From field of Double-Take generated e-mail messages.

Values—Any valid e-mail address, up to 256 characters

Default—None

Console Setting—Edit Server Properties page, E-mail Notification section, From address

Service restart required—No

EmailIncludeCategories

Description—Specifies which Event Viewer messages are sent via e-mail

Values—1 Error messages will be sent via e-mail, 2 Warning messages will be sent via e-mail, 3 Information messages will be sent via e-mail

Default—1,2

Console Setting—Edit Server Properties page, E-mail Notification section, Include these events

Service restart required—Yes from the registry, No from the console

EmailNotificationList

Description—Specifies the e-mail address(es) that will receive Double-Take generated e-mail messages.

Values—A comma separated list of valid e-mail addresses, up to 256 addresses. Each address is limited to 256 characters.

Default—None

Console Setting—Edit Server Properties page, E-mail Notification section, Send to

Service restart required—No

EmailPassword

Description—The password required for SMTP server authentication

Values—Any valid password text

Default—None

Console Setting—Edit Server Properties page, E-mail Notification section, Password

Service restart required—No

Notes—Since the password is encrypted for security, this entry cannot be displayed or changed through the registry.

EmailServer

Description—The name of the SMTP server for e-mail notification

Values—Any valid server name text

Default—None

Console Setting—Edit Server Properties page, E-mail Notification section, E-mail server (SMTP)

Service restart required—No

EmailServerPort

Description—Specifies the port that the SMTP e-mail server is using

Values—any valid port number

Default—25

Console Setting—None

Service restart required—No

EmailSmtpLogin

Description—Specifies if SMTP server authentication for e-mail notification is enabled or disabled

Values—0 SMTP authentation is disabled, 1 SMTP authenticaion is enabled

Default—0

Console Setting—Edit Server Properties page, E-mail Notification section, Log on to e-mail server

Service restart required—No

Notes—Your SMTP server must support the LOGIN authentication method to use this feature. If your server supports a different authentication method or does not support authentication, you may need to add the Double-Take server as an authorized host for relaying e-mail messages. This option is not necessary if you are sending exclusively to e-mail addresses that the SMTP server is responsible for.

EmailSubjectDesc

Description—Specifies if the Event Viewer message will be appended to the end of the subject line for e-mail notification

Values—0 Event Viewer message is not included in the subject line, 1 Event Viewer message is included in the subject line

Default—1

Console Setting—Edit Server Properties page, E-mail Notification section, Add event description to subject

Service restart required—No

EmailSubjectPrefix

Description—Specifies unique text which will be inserted at the front of the subject line for each Double-Take generated e-mail message. This will help distinguish the Double-Take messages from other messages.

Values—Any valid text

Default—Double-Take Notification

Console Setting—Edit Server Properties page, E-mail Notification section, Subject prefix

Service restart required—No

EmailUsername

Description—The user ID required for SMTP server authentication

Values—Any valid user ID text

Default—None

Console Setting—Edit Server Properties page, E-mail Notification section, User name

Service restart required—No

Notes—Since the username is encrypted for security, this entry cannot be displayed or changed through the registry.

EnableCRCCheck

Description—Indicates if Double-Take will perform a cyclic redundancy check between the source and target to identify corrupted packets

Values—0 Disabled, 1 Enabled

Default—0

Console Setting—None

Service restart required—No

Notes—This option only needs to be set on the source server. However, if you will be restoring or reversing, where the roles of the servers are reversed, then you will need to set this option on the target as well.

EnableDHCP

Description—Indicates if Double-Take DHCP support is enabled

Values—0 Disabled, 1 Enabled

Default—1

Console Setting—None

Service restart required—No

EnableEFSVerify

Description—Indicates if Double-Take will verify Microsoft encryption on the source before transmitting the encrypted file to the target

Values—0 Disabled, 1 Enabled

Default—0

Console Setting—None

Service restart required—No

EnableFileOpenTracing

Description—Specifies if debug-level messages are enabled to trace all mirroring and replicated files that are opened

Values—0 Do not trace files that are opened, 1 Trace files that are opened

Default—0

Console Setting—None

Service restart required—Yes

Notes—This option should only be enabled (1) for temporary, debug sessions as instructed by technical support.

EnablePerformanceTracking

Description—This entry will be used in the future.

EnableRootEncryption

Description—Specifies if the top-level folders of a protected data set are encrypted on the source, they will be encrypted on the target as well

Values—0 Disabled, 1 Enabled

Default—1

Console Setting—None

Service restart required—No

Notes—If the top-level folders in a protected data set are not encrypted, disabling this option may obtain a small performance improvement.

EnableShortFileNameProcessing

Description—Indicates if Double-Take will correct any short file names created by the operating system on the target during a mirror or for create and rename operations during replication

Values—0 Do not correct any short file names on the target, 1 Correct short file names on the target

Default—0

Console Setting—None

Service restart required—No

EnableSnapshots

Description—Specifies whether Double-Take snapshot functionality is enabled

Values—0 Double-Take snapshot functionality is disabled, 1 Double-Take snapshot functionality is enabled

Default—1

Console Setting—None

Service restart required—Yes

Notes—This setting only impacts Double-Take snapshot functionality. If this setting is disabled, other snapshot software such as Microsoft Volume Shadow Copy will be not be impacted.

EnableTaskCmdProcessing

Description—Queues tasks inline with replication data

Values—0 Disable task command processing, 1 Enable task command processing

Default—0

Console Setting—Edit Server Properties page, Setup section, Enable task command processing

Service restart required—No

EncryptNetworkData

Description—Encrypts Double-Take data before it is sent from the source to the target

Values—0 Disable data encryption, 1 Enable data encryption

Default—0

Console Setting—None

Service restart required—No

Notes—Both the source and target must be Double-Take encryption capable (Double-Take version 7.0.1 or later), however this option only needs to be enabled on the source or target in order to encrypt data. Keep in mind that all jobs from a source with this option enabled or to a target with this option enabled will have the same encryption setting. Changing this option will cause jobs to auto-reconnect and possibly remirror.

ExtensionNumber

Description—Used by the Double-Take log files.

FailbackHostname

Description—Returns the host SPN (Service Principle Name) to its original setting on failback

Values—0 Disabled, 1 Enabled

Default—0

Console Setting—None

Service restart required—No

Notes—If you are using Active Directory, this option should be enabled or you may experience problems with failback.

FailoverData1

Description—An internal setting for failover. Do not modify this setting.

FailoverData2

Description—An internal setting for failover. Do not modify this setting.

FailoverHostname

Description—Automatically removes the host SPN (Service Principle Name) from Active Directory on the source

Values—0 Disabled, 1 Enabled

Default—0

Console Setting—None

Service restart required—No

Notes—If you are using Active Directory, this option should be enabled or you may experience problems with failover.

FailoverOnRouteFailure

Description—Determines if failover will occur when receiving a router message back from an IP address on the network

Values—0 Failover will not occur when receiving a destination host unreachable message, 1 Failover will occur when receiving a destination host unreachable message

Default—1

Console Setting—None

Service restart required—No

FCCHelpPath

Description—This entry is no longer used.

FileAccessRetry

Description—The number of times a failed driver call will be retried by the service.

Values—1 - 65535

Default—10

Console Setting—None

Service restart required—No

FileQueueSize

Description—When a mirror is started, one thread reads from the disk and builds the file queue. Another set of threads reads files off of the queue and sends them to the target. This setting is the maximum size of the queue in entries. If you had 100 files to be mirrored and this was set to 16 (the default value), the first thread would fill the queue to a maximum of 16 entries.

Values—1 - 65535

Default—16

Console Setting—None

Service restart required—No

Notes—This value must be set prior to starting the mirror process. The higher the number, the more memory that is used.

ForceReplaceOnFailover

Description—Specifies additional failover options

Values—0 Use standard failover add / replace settings with no additional settings, 1 Replace the target server name with that of the source and add the source IP address, 2 Add the source server name to the target and replace the target IP address, 3 Replace the target server name with that of the source and replace the target IP address

Default—0

Console Setting—None

Service restart required—No

ForceVerifyOnMirror

Description—Specifies if verification will be performed with every difference mirror

Values—0 Verification is not performed with every difference mirror, 1 Verification is performed with every difference mirror

Default—0

Console Setting—None

Service restart required—No

GenerateDumpOnShutdownCrashes

Description—Specifies if a log file will be created during a shutdown crash

Values—0 No log file is created and the default exception handler is used, 1 Log file is created

Default—0

Console Setting—None

Service restart required—Yes

HardLinkInterval

Description—Specifies the length of time, in seconds, to generate a hard link report

Values—any valid integer

Default—3600

Console Setting—None

Service restart required—No

HardLinkLogPath

Description—Specifies the location where hard links will be logged. If no path is specified, the location defined in LogDir will be used.

Values—any valid path

Default—None

Console Setting—None

Service restart required—No

HBLoopback

Description—This entry is no longer used.

HBTrace

Description—Specifies whether heartbeat debugging information is generated

Values—0 not generated, 1 Generated

Default—0

Console Setting—None

Service restart required—No

HBTTL

Description—Number of seconds without receiving a heartbeat before a remote machine is considered unavailable

Values—0 - 65535

Default—10

Console Setting—None

Service restart required—No

HeartbeatIgnoreIPs

Description—This setting is no longer used.

HPQueueRatio

Description—Ratio of replication packets to one mirror packet

Values—1 - 65535

Default—5

Console Setting—Edit Server Properties page, Source section, Number of replication packets per one mirror packet

Service restart required—No for future connections, Yes for the current connection

Notes—An HPQueueRatio of 5 allows Double-Take to dynamically change the ratio as needed based on the amount of replication data in queue. If you set a specific value other than the default (other than 5), the specified value will be used.

IgnoreAlternateStreamFiles

Description—Specifies alternate streams to skip during mirroring and replication

Values—a semi-colon separate list of stream names. The stream names are not case-sensitive

Default—none

Console Setting—None

Service restart required—No

IgnoreArchiveBit

Description—Specifies if the archive bit is compared during verification

Values—0 Archive bit is compared during a verification, 1 Archive bit is not compared during a verification

Default—1

Console Setting—None

Service restart required—No

IgnoreDeleteOps

Description—Specifies if file and directory delete operations will be replicated to the target

Values—0 Delete operations are replicated to the target, 1 Delete operations are not replicated to the target

Default—0

Console Setting—None

Service restart required—No

IgnoreOpLockErrors

Description—Specifies how files that are locked open on the source are handled during mirroring

Values—0 Fail the mirror and record OpLock errors in the log. The job state will be set to mirror required, 1 Ignore the lock errors and continue the mirror. This option does not guarantee data integrity. There may be differences in the file that was locked.

Default—0

Console Setting—None

Service restart required—No

IgnorePPPAddresses

Description—Identifies if Double-Take will use PPP (Point-to-Point Protocol) or SLIP (Serial Line Internet Protocol) adapters

Values—0 Double-Take will send out heartbeats across the PPP/SLIP adapter, 1 Double-Take will not send out heartbeats across the PPP/SLIP adapter

Default—1

Console Setting—None

Service restart required—No

IgnoreSourceErrors

Description—Ignores source errors that will cause an update to the target data state

Values—0 Do not ignore source errors, 1 Ignore source errors

Default—0

Console Setting—None

Service restart required—No

IgnoreThumbnailStreams

Description—Specifies if thumbnails will be replicated to the target.

Values—0 Double-Take will mirror and replicate all data streams, 1 Double-Take will not mirror or replicate any data about the alternate data streams for thumbnail images. When comparing data for a verification or difference mirror,alternate data streams for thumbnails will not be reported as different.

Default—1

Console Setting—None

Service restart required—If you change this value to 0, you must restart the Double-Take service in order for the Double-Take driver to begin sending all data stream information to the service. If you change this value to 1, you do not need to restart the service.

IgnoreWriteFailureOnTarget

Description—Specifies whether failures to write a file on the target are logged

Values—0 Log all write failures on the target, 1 or any larger integer indicates that number of write failures which will be ignored before starting to log the write failures

Default—0

Console Setting—None

Service restart required—No

IncludeSysVolInfo

Description—Specifies whether the system volume information folder is mirrored and replicated

Values—0 Do not include the system volume information folder, 1 Include the system volume information folder

Default—0

Console Setting—None

Service restart required—No

InstallPath

Description—Path specified during the Double-Take installation. Do not modify this entry.

InstallVersionInfo

Description—Installation number specified during the Double-Take installation. Do not modify this entry.

InteractWithDesktopForFOScripts

Description—Runs the failover scripts interactively on the user's local session

Values—0 Do not run the scripts interactively, 1 Run the scripts interactively.

Default—0

Console Setting—None

Service restart required—Yes

IntermediateQueueLimit

Description—Amount of memory, in KB, that may be allocated to the intermediate queue by the system memory manager when MemoryAllocatorMode is set to mixed mode (2).

Values—512-4194304

Default—65536

Console Setting—None

Service restart required—Yes

IPFailover

Description—Specifies whether or not to failover the IP addresses during failover

Values—0 Do not failover IP addresses 1 Failover IP addresses

Default—1

Console Setting—Set Options page, Failover Options section, Failover IP addresses

Service restart required—No

KFAIOpenRetry

Description—Specifies the number of times an operation is retried if the driver return an error

Values—any valid integer

Default—10

Console Setting—None

Service restart required—No

LanguagesAvailable

Description—Specifies the Double-Take language support that has been installed. Do not modify this setting. If you need to add or remove language support, use the Double-Take installation program.

LanguageSelected

Description—Specifies the language of the verification log

Values—Depends on LanguagesSupported

Default—Language used during the installation

Console Setting—Edit Server Properties page, Logging section, Language

Service restart required—Yes

LanguagesSupported

Description—Specifies the available languages for the verification log. Currently English is the only language available. Do not modify this setting.

LastModifiedReadDelay

Description—Specifies the length of time, in seconds, to wait before reading the last modified file time attribute

Values—any valid integer

Default—15

Console Setting—None

Service restart required—No

Notes—This option is only used if SendLastModifiedTimeOnClose is disabled

LoadSourceTarget

Description—Specifies the functionality of the Double-Take service

Values—0 Neither the source nor target modules are loaded, 1 Only the source module is loaded, 2 Only the target module is loaded, 3 Both the source and target modules are loaded

Default—3

Console Setting—None

Service restart required—Yes

LogAllOrphans

Description—This entry is no longer used.

LogDir

Description—The location of the Double-Take messages/alerts, verification, and statistics log files

Values—any valid path

Default—the location where the Double-Take files were installed

Console Setting—Edit Server Properties page, Logging section, Logging folder

Service restart required—Yes

LogFile

Description—The name of the Double-Take messages/alerts log file

Values—any valid file name

Default—DTLog

Console Setting—None

Service restart required—No

LogHardlinks

Description—Indicates whether hard links are logged to replication_set_name.log when the protected data set size is calculated

Values—0 Hard links are not logged, 1 Hard links are logged

Default—0

Console Setting—None

Service restart required—No

LogMessageLevel

Description—Specifies the types of messages logged to the.dtl files

Values—0 No messages will be logged, 1 Only alert messages will be logged, 2 Alert and release messages will be logged, 3 Alert, release, and debug messages will be logged

Default—2

Console Setting—None

Service restart required—No

LoopbackID

Description—This entry is no longer used.

MaxChecksumBlocks

Description—Specifies the number of checksum values retrieved from the target

Values—any integer

Default—32

Console Setting—None

Service restart required—No

MaxConnections

Description—Number of network requests that can be processed simultaneously. Windows is limited to 5 simultaneous requests.

Values—0 - 65535

Default—5

Console Setting—None

Service restart required—Yes

Notes—Vision Solutions recommends that you not change this value.

MaxLogFileSize

Description—Maximum size, in bytes, of any .dtl log file

Values—limited by available disk space

Default—5242880

Console Setting—Edit Server Properties page, Logging section, Maximum size (under Messages & Alerts)

Service restart required—No

MaxLogPathname

Description—The maximum length of a file name (the entire volume\directory\filename including slashes, spaces, periods, extensions, and so on) that will be displayed in the Double-Take log file and the Windows Event Viewer. File names longer than the MaxDisplayablePath will be truncated and will be followed by an ellipsis (...).

Values—1-32760

Default—32760

Console Setting—None

Service restart required—No

MaxNumberofLogFiles

Description—Maximum number of .dtl log files that can exist at one time. When Double-Take creates a new .dtl file, if this number is exceeded, the oldest .dtl file is deleted.

Values—1 - 999

Default—20

Console Setting—Edit Server Properties page, Logging section, Maximum number of files

Service restart required—No

MaxOpBufferSize

Description—An internal setting for memory buffering. Do not modify this setting.

MaxRemoveOrphansOpSize

Description—Determines whether or not Double-Take will send over multiple orphan operations. Double-Take will send over the operations if a directory has more files than this number.

Values—0 - 131072

Default—1000

Console Setting—None

Service restart required—No

MaxRetry

Description—A generic, application wide setting specifying the number of retry attempts for processes such as creating sockets, starting the service, and so on

Values—any integer

Default—5

Console Setting—None

Service restart required—Yes

MaxWriteChunkSize

Description—Maximum merged op size (in bytes) used during replication

Values—1 - 131072

Default—65536

Console Setting—None

Service restart required—No

MCHelpPath

Description—This entry is no longer used.

MemoryAllocatorCallbackMode

Description—Determines what action is taken when the MemoryQueueToDiskThreshold is met

Values—0 Auto-disconnect processing is initiated when theMemoryQueueToDiskThreshold has been met. Connections will be reestablished when auto-reconnect occurs, 1 The Double-Take service stops pulling operations from the driver when theMemoryQueueToDiskThreshold has been met. The target will pause the source.The service will resume pulling operations when the target tells the source to resume, 2 The source and target begin queuing operations to disk.

Default—2

Console Setting—None

Service restart required—Yes

MemoryQueueToDiskThreshold

Description—A percentage of QmemoryBufferMax that will trigger queuing to disk.

Values—any valid percentage

Default—75

Console Setting—None

Service restart required—Yes

MinCompressionFileSize

Description—The minimum file size, in bytes, that will be compressed. Files smaller than this size will not be compressed.

Values—any file size

Default—1024

Console Setting—None

Service restart required—No

MirrorChunkSize

Description—Block size, in bytes, used in the mirroring process

Values—1 - 1048576

Default—65536

Console Setting—Edit Server Properties page, Source section, Size of mirror packets

Service restart required—No

Notes—A higher block size value gives you better throughput, but only to a certain point, then it starts using more memory (this has to do with the way memory is allocated and deallocated). A lower block size value produces slower throughput, but uses memory efficiently.

MirrorEncryptedFiles

Description—Specifies if Windows 200x encrypted files are mirrored

Values—0 Encrypted files are not mirrored, 1 Encrypted files are mirrored

Default—1

Console Setting—None

Service restart required—No

MirrorOverwrite

Description—Determines if the mirror process overwrites existing files

Values—0 never overwrite, 1 always overwrite, 2 overwrite if older

Default—1

Console Setting—None

Service restart required—No

MirrorPrompting

Description—This entry is no longer used.

MirrorQueueLimit

Description—Maximum number of mirror operations that can be queued on the source machine

Values—1 - 65535

Default—1000

Console Setting—Edit Server Properties page, Source section, Maximum pending mirror operations

Service restart required—No

MirrorRootAttributes

Description—Specifies whether or not root permissions from the source are mirrored to the target

Values—0 Root permissions are not mirrored to the target, 1 Root permissions are mirrored to the target

Default—1

Console Setting—None

Service restart required—No

MirrorZeroKFiles

Description—Specifies whether or not empty files, zero byte files, are included in a mirror

Values—0 Zero byte files are skipped and not mirrored to the target, 1 All files are mirrored to the target

Default—1

Console Setting—None

Service restart required—No

Notes—If MirrorZeroKFiles is enabled (0), zero byte files are skipped during a full mirror, file differences mirror, and a verification with synchronization. Zero byte files that contain alternate data streams that are not empty, will still be skipped if MirrorZeroKFiles is enabled.

MissedHeartbeats

Description—Specifies the number of heartbeats that can go unanswered by the source before failover occurs, when using Double-Take service responses to monitor for failover

Values—1 - 65535

Default—20

Console Setting—None

Service restart required—No

Notes—This value is used in conjunction with the PingFrequency value to determine the value of Consider the source server failed (on the Set Options page, Failover Monitor section).

MissedPackets

Description—Specifies the number of requests sent by the target that go unanswered by the source before failover occurs, when using network responses to monitor for failover

Values—1 - 65535

Default—5

Console Setting—None

Service restart required—No

Notes—This value is used in conjunction with the PingFrequency value to determine the value of Consider the source server failed (on the Set Options page, Failover Monitor section).

MoveOrphanedFiles

Description—This entry is no longer used.

MoveOrphansDir

Description—This entry is no longer used.

NameFailover

Description—Specifies whether or not to failover machine names

Values—0 Do not failover machine names, 1 Failover machine names

Default—1

Console Setting—Set Options page, Failover Options section, Failover server name

Service restart required—No

NetPort

Description—Port used by pre-5.2 versions for TCP communications

Values—1025 - 65535

Default—1100

Console Setting—None

Service restart required—Yes

NetworkRetry

Description—Specifies the interval, in seconds, at which Double-Take will attempt to reconnect to the target

Values—any positive number

Default—10

Console Setting—None

Service restart required—No

NetworkStatusInterval

Description—An internal setting for network communications. Do not modify this setting.

NetworkTimeout

Description—The maximum length of time, in seconds, to wait on a network connection. If data is not received over a network connection within the specified time limit, the connection is closed. During idle periods, Double-Take sends small amounts of keep-alive data at an interval 1/6 of the NetworkTimeout value to keep the socket from being inadvertently closed.

Values—any integer

Default—120

Console Setting—None

Service restart required—No

Notes—If you are archiving files and it takes longer than the NetworkTimeout specified (for example, this may happen if the DTArchiveBin is located on an alternate volume), the archive operation will complete on the target, but the full file will not be changed to a link on the source because the source detected the network timeout.

NodeLockedLicenseKey

Description—An internal setting for licensing. Do not modify this setting.

NodeLockedServerInfo

Description—An internal setting for licensing. Do not modify this setting.

OpBufferMax

Description—Specifies the number of operations that can be stored in the memory queue prior to queuing to disk

Values—0 There is no limit to the number of operations that can be stored in the memory queue, 1 or any larger integer

Default—200000

Console Setting—None

Service restart required—No

OpBuffersCount

Description—An internal setting for memory buffering. Do not modify this setting.

OpLogging

Description—Specifies whether operations from the Double-Take driver are logged

Values—0 Do not log operations, 1 Log operations

Default—0

Console Setting—None

Service restart required—Yes

OutOfOrderDiff

Description—The maximum number of operations that can be out of order before the connection is paused

Values—any integer

Default—10

Console Setting—None

Service restart required—No

Notes—The larger the value, the more memory the Double-Take service on the targe service will use.

PingFrequency

Description—Specifies, in seconds, how often a ping is sent to the source from a monitoring target

Values—1 - 65535

Default—5

Console Setting—None

Service restart required—No

Notes—This value is used in conjunction with the MissedHeartbeats or MissedPackets value to determine the value of Consider the source server failed (on the Set Options page, Failover Monitor section).

Port

Description—Port connection for core Double-Take communications

Values—1025 - 65535

Default—6320

Console Setting—Edit Server Properties page, General section, Port

Service restart required—Yes

PostFailbackScript

Description—Location on the target where the target post-failback script is located

Values—Any valid path

Default—<null>

Console Setting—Set Options page, Failover Options section, Script file (under Post-failback script)

Service restart required—No

PostFailbackScriptArgs

Description—Arguments to be used with the target post-failback script

Values—Any valid argument

Default—<null>

Console Setting—Set Options page, Failover Options section, Arguments (under Post-failback script)

Service restart required—No

PostFailoverScript

Description—Location on the target where the target post-failover script is located

Values—Any valid path

Default—<null>

Console Setting—Set Options page, Failover Options section, Script file (under Post-failover script)

Service restart required—No

PostFailoverScriptArgs

Description—Arguments to be used with the target post-failover script

Values—Any valid argument

Default—<null>

Console Setting—Set Options page, Failover Options section, Arguments (under Post-failback script)

Service restart required—No

PreFailbackScript

Description—Location on the target where the target pre-failback script is located

Values—Any valid path

Default—<null>

Console Setting—Set Options page, Failover Options section, Script file (under Pre-failback script)

Service restart required—No

PreFailbackScriptArgs

Description—Arguments to be used with the target pre-failback script

Values—Any valid argument

Default—<null>

Console Setting—Set Options page, Failover Options section, Arguments (under Pre-failback script)

Service restart required—No

PreFailbackWait

Description—Specifies whether or not to wait for the target pre-failback script to complete before finishing a failback

Values—0 Do not wait, 1 Wait

Default—0

Console Setting—Set Options page, Failover Options section, Delay until script completes (under Pre-failback script)

Service restart required—No

PreFailoverScript

Description—Location on the target where the target pre-failover script is located

Values—Any valid path

Default—<null>

Console Setting—Set Options page, Failover Options section, Script file (under Pre-failover script)

Service restart required—No

PreFailoverScriptArgs

Description—Arguments to be used with the target pre-failover script

Values—Any valid argument

Default—<null>

Console Setting—Set Options page, Failover Options section, Arguments (under Pre-failover script)

Service restart required—No

PreFailoverWait

Description—Specifies whether or not to wait for the target pre-failover script to complete before finishing a failover

Values—0 Do not wait, 1 Wait

Default—0

Console Setting—Set Options page, Failover Options section, Delay until script completes (under Pre-failover script)

Service restart required—No

ProductCode

Description—Used by the Double-Take installation program to maintain the installation settings for an upgrade. Do not modify this entry.

ProductName

Description—Used by the Double-Take installation program to maintain the installation settings for an upgrade. Do not modify this entry.

QJournalDir

Description—The location where the queue is stored.

Values—any valid path

Default—the location specified during the installation

Console Setting—Edit Server Properties page, Queue section, Queue folder

Service restart required—No

Notes—For best results and reliability, you should select a dedicated, non-boot volume. The queue should be stored on a fixed, local NTFS volume. This location also stores the Double-Take driver pagefile.

QJournalFileSize

Description—The size, in MB, of each queuing transaction log file.

Values—any valid file size, up to 4095 MB

Default—5

Console Setting—None

Service restart required—No

QJournalFreeSpaceMin

Description—The minimum amount of disk space, in MB, in the specified QJournalDir that must be available at all times.

Values—dependent on the amount of physical disk space available

Default—250

Console Setting—Edit Server Properties page, Queue section, Minimum free disk space

Service restart required—No

Notes—The QJournalFreeSpaceMin should be less than the amount of physical disk space minus QJournalSpaceMax.

QJournalPreload

Description—The number of operations being pulled from the disk queue at one time. Do not modify this setting.

QJournalSpaceMax

Description—The maximum amount of disk space, in MB, in the specified QJournalDir that can be used for Double-Take queuing. When this limit is reached, Double-Take will automatically begin the auto-disconnect process.

Values—dependent on the amount of physical disk space available

Default—Unlimited

Console Setting—Edit Server Properties page, Queue section, Limit disk space for queue

Service restart required—No

Notes—The unlimited setting allows the disk queue usage to automatically expand whenever the available disk space expands. Setting this option to zero (0) disables disk queuing. Even if you are using the unlimited option, Double-Take will only store 16,384 log files. If you are using the default 5MB file size, this is approximately 80GB of data. If you anticipate needing to be able to queue more data than this, you should increase the size of the log files.

QLogWriteThrough

Description—Specifies if the disk queues are write-through mode

Values—0 Disk queues are not write-through mode, 1 Disk queues are write-through mode

Default—0

Console Setting—None

Service restart required—No

Notes—While write-through mode may decrease the frequency of auto-disconnects, it may also decrease the performance of the source server.

QMemoryBufferMax

Description—The amount of Windows system memory, in MB, that, when exceeded, will trigger queuing to disk.

Values—minimum 512, maximum is dependent on the server hardware and operating system

Default—1024

Console Setting—Edit Server Properties page, Queue section, Amount of system memory to use

Service restart required—Yes

QueryOnQuorumFile

Description—Identifies if the Double-Take service will reopen closed files on the quorum drive

Values—0 The Double-Take service will not attempt to reopen a closed file on the quroum drive to get security descriptors or last modified times, 1 The Double-Take service will attempt to reopen a closed file on the quroum drive to get security descriptors or last modified times.

Default—1

Console Setting—None

Service restart required—No

QueueSizeAlertThreshold

Description—The percentage of the queue that must be in use to trigger an alert message in the Windows Event Viewer.

Values—any valid percentage

Default—50

Console Setting—Edit Server Properties page, Queue section, Alert at this queue usage

Service restart required—Yes

Registered

Description—This entry is no longer used.

RemoveAllOrphans

Description—This entry is no longer used.

RemoveOrphansTime

Description—This entry is no longer used.

RemoveSharesOnDisconnect

Description—Specifies if shares are removed on the target machine when a Double-Take protected data set is disconnected from a target or a source machine is manually shutdown by the administrator. (Shares are not removed if either the source or target machines fail.)

Values—0 Remove shares from the target, 1 Do not remove shares from the target

Default—1

Console Setting—None

Service restart required—No

ReplaceTarget

Description—Specifies whether or not to replace the target identity with the source identity during a failover

Values—0 Do not replace, 1 Replace

Default—0

Console Setting—None

Service restart required—No

ReplicateNtSecurityByName

Description—Determines whether or not Double-Take replicates permissions and attributes assigned to local (non-domain) users and groups

Values—0 Do not replicate by name, 1 Replicate by name

Default—0

Console Setting—Edit Server Properties page, Source section, Replicate NTFS security attributes by name

Service restart required—No

ReplicationDiskCheckScript

Description—Specifies the script to run if validation of the replication drive fails

Values—Any valid path and script file

Default—<null>

Console Setting—None

Service restart required—No

ReplicationDiskCheckTimeOut

Description—Specifies the interval, in seconds, between validation checks when ReplicationDiskCheckSript is populated

Values—any integer

Default—300

GUI Setting—None

Service restart required—No

RepSetDBName

Description—Name of the database that contains protected data set information

Values—any valid file name

Default—DblTake.db

Console Setting—None

Service restart required—No

RestoreOverwrite

Description—Determines if the restoration process overwrites existing files

Values—0 never overwrite, 1 always overwrite, 2 overwrite if older

Default—2

Console Setting—None

Service restart required—No

RestorePrompting

Description—This entry is no longer used.

RetentionFlag

Description—This entry will be used in the future.

RunDTInfoOnCutover

Description—Specifies if DTInfo is launched before a failover or cutover when protecting an entire server

Values—0 Do not launch DTInfo, 1 Launch DTInfo

Default—1

Console Setting—None

Service restart required—No

RunScriptatSnaptime

Description—If a script is specified, the script is launched on the target before Double-Take executes any snapshots. The snapshot will not be executed until the script has completed. If the script returns an error, the snapshot will still execute.

Values—any valid path and script name

Default—<null>

Console Setting—None

Service restart required—No

RunScriptInsteadofSnap

Description—Specifies if a script specified in RunScriptAtSnaptime is executed

Values—0 Execute script specified in RunScriptAtSnaptime, 1 Do not execute script specified in RunScriptAtSnaptime

Default—1

Console Setting—None

Service restart required—No

SaveStatFile

Description—Determines if the statistic.sts (statistics logging) file is saved or ovewritten

Values—0 overwrite, 1 saved as statistic-old.sts

Default—1

Console Setting—None

Service restart required—No

ScheduleFile

Description—Name of the database file that contains transmission scheduling information

Values—any valid file name

Default—Schedule.sts

Console Setting—None

Service restart required—Yes

ScheduleInterval

Description—The number of seconds to wait before checking the transmission schedules to see if transmission should be started or stopped

Values—1 - 3600

Default—1

Console Setting—None

Service restart required—Yes

SendDirLastModifiedTime

Description—Specifies if the last modified time for directories will be transmitted to the target during a difference mirror

Values—0 last modified time on directories will not be sent to the target, 1 last modified time on directories will be sent to the target

Default—1

Console Setting—None

Service restart required—No

SendFileTimesOnCreate

Description—Specifies whether a file is accessed twice so that the file's creation time can be modified to match the source

Values—0 The Double-Take service will not access newly created files that have not been modified. These files on the target will have the date and time of when the file was created on the target, 1 The Double-Take service will access newly created files. These files on the target will have the same date and time as the source.

Default—0

Console Setting—None

Service restart required—No

Notes—New files created on the source that have not been modified will have the date and time of when the file is created on the target. The date and time will be corrected to match the source's true file attributes when a remirror or verification modifies them to match the source or the file is modified by a user or application on the source. For example, if the source machine's clock is set to 2:00 PM and the target machine is set to 4:00 PM, a newly created file that has not been modified will have a time stamp of 4:00 PM when it is applied to the target. If this option is enabled (1), Double-Take will access the file twice, to correctly set the time to 2:00 PM to reflect the file's true attributes. If this option is disabled (0), Double-Take will not access the file twice, and the file will have the target time of 4:00 PM until it is modified (remirror, verification, or user or application update).

SendLastModifiedTimeOnClose

Description—Specifies that the last modified time attribute is sent when a file is closed

Values—0 Last modified time is sent when Double-Take has not received any additional operations for the file in the time period specified by LastModifiedReadDelay, 1 Last modified time is sent when a file is closed, which may not be immediately depending on system processing

Default—1

Console Setting—None

Service restart required—No

Notes—If system processing delays (such as the system cache manager not flushing quickly enough) are causing delays in processing the last modified time, you may want to consider disabling this option (0).

ServerUUID

Description—Used internally by the Double-Take service to identify Double-Take connections and IP addresses used between servers

Values—Unique identifier generated by Double-Take

Default—Generated by Double-Take

Console Setting—None

Service restart required—Yes

Notes—If you are certain that the server is not being used by any jobs, you can delete the ServerUUID. For example, you may want to delete the ServerUUID so that you can create an image of a server after installing Double-Take. A deleted ServerUUID will be re-created the next time the Double-Take service is started. Keep in mind, if you delete the ServerUUID and the server is being used by any jobs, you will have problems with all aspects of Double-Take including mirroring, replication, and failover.

ServicePriority

Description—The priority level at which the Double-Take service runs.

Values—2 normal priority, 3 high priority

Default—2

Console Setting—None

Service restart required—Yes

Notes—The Double-Take service runs at normal priority by default. This option should not be modified, however, if the priority is raised to high (3), it can be done through Windows Task Manager.

ServicesToKeepRunning

Description—Services that will not be stopped on the target

Values—Semi-colon separated list of service names

Default—<null>

Console Setting—Set Options page, Target Services section, Services to leave running on the target server during protection

Service restart required—No

Notes—You can specify the service name using the service executable file name or the service display name. There is no need to use quotation marks, even if the names have spaces in them. Only separate the names by a semi-colon (;).

ServiceStopState

Description—Used internally by the Double-Take service. Do not modify this entry.

ShareFailover

Description—Specifies whether or not to failover shares

Values—0 Do not failover shares, 1 Failover shares

Default—1

Console Setting—Set Options page, Failover Options section, Failover shares

Service restart required—No

ShareUpdateInterval

Description—Specifies how often, in minutes, the share file will be sent to the target

Values—1 - 65535

Default—60

Console Setting—None

Service restart required—No

ShortFileNameScanIntervalMinutes

Description—Specifies how often, in minutes, the registry is scanned for short file names

Values—any valid integer

Default—240

Console Setting—None

Service restart required—No

ShutdownRebootTimeoutMinutes

Description—Specifies the amount of time, in minutes, to wait for the source to shutdown during failover or cutover

Values—any valid integer

Default—5

Console Setting—None

Service restart required—No

ShutdownTimeout

Description—The amount of time, in seconds, for the service to wait prior to completing the shutdown so that Double-Take can persist data on the target in an attempt to avoid a remirror when the target comes back online

Values—any valid number of seconds where 0 (zero) indicates waiting indefinitely and any other number indicates the number of seconds

Default—0

Console Setting—Edit Server Properties page, Setup section, Time allowed to complete shutdown operations

Service restart required—No

Notes—This setting only controls the service shutdown from the Double-Take clients. It does not control the service shutdown through a reboot or from the Service Control Manager.

SkipCompressionFileExt

Description—A period delimited list of file types that are not compressed, even if compression is enabled.

Values—any period delimited list of file types

Default—mp3.exe.wmv.wma.qt.mpg.mpeg.zip.jpg.jpeg.tiff.tar.rar.cab

Console Setting—None

Service restart required—No

SnapshotType

Description—Specifies the type of snapshot that Double-Take takes

Values—0 Create a client-accessible or non-client-accessible snapshot based on the job type , 1 Always create a client-accessible snapshot, 2 Always create a non-client-accessible snapshot

Default—0

Console Setting—None

Service restart required—No

SourceNewerMaxFileCount

Description—The number of files to compare during a source newer difference mirror

Values—1-1000

Default—16

Console Setting—None

Service restart required—No

SourcePendingAcks

Description—The number of operations received by the target queue in which the source is waiting for a response

Values—100 - 20,000

Default—2000

Console Setting—None

Service restart required—No

SourcePostFailbackScript

Description—Path on the source where the source post-failback script is located

Values—Any valid path

Default—<null>

Console Setting—None

Service restart required—No

SourcePostFailbackScriptArgs

Description—Arguments to be used with the source post-failback script

Values—Any valid argument

Default—<null>

Console Setting—None

Service restart required—No

SSMKeepTargetActivationCode

Description—Specifies if the activation code on the target is replaced or maintained after a full-serer failover or cutover. Do not modify this entry.

SSMShutdownServices

Description—Used by full server jobs to determine services to shutdown during failover or cutover. Do not modify this entry.

SSMShutdownSource

Description—Specifies if the source is shutdown when performing failover or cutover

Values—0 The source will not be shutdown, 1 The source will be shutdown

Default—1

Console Setting—Set Options page, Failover Options section, Shutdown the source server

Service restart required—Yes

Notes—This setting must be applied on the target server.

SSMStagingBase

Description—Specifies the folder to use for staging system state files for full server failover or cutover. Do not modify this entry.

SSMUseDiskSignature

Description—Used by full server jobs to determine how target disk signatures are used. Do not modify this entry.

StartupScript

Description—Used by full server jobs to control the post-failover script after reboot after failover. Do not modify this entry.

StatsDriverLogFlags

Description—Indicates which driver statistics are logged to the Double-Take log

Values—0 No driver statistics are logged, 1 State, 2 Operations, 4 Paging, 8 Timing

Default—0

Console Setting—None

Service restart required—Yes

Notes—Use the sum of various values to log multiple driver statistics. For example, a setting of 5 would log paging and state statistics. A setting of 7 would log paging, operations, and state statistics. A setting of 15 would log all driver statistics.

StatsFileName

Description—Default file for logging statistics

Values—any valid file name

Default—statistic.sts

Console Setting—Edit Server Properties page, Logging section, Filename (under Statistics)

Service restart required—No

StatsLoggingOn

Description—Specifies if Double-Take logs statistics at startup

Values—0 Stats logging does not start when Double-Take starts, 1 Stats logging starts when Double-Take starts

Default—0

Console Setting—Edit Server Properties page, Setup section, Setup Options, Log statistics automatically

Service restart required—No

StatsMaxFileSize

Description—Maximum size, in MB, for the statistic.sts file

Values—limited by available disk space

Default—10485760

Console Setting—Edit Server Properties page, Logging section, Maximum size (under Statistics)

Service restart required—No

StatsMaxObjects

Description—This entry is no longer used.

StatsPort

Description—Port used by pre-5.2 versions for DTStat to gather statistics

Values—1025 - 65535

Default—1106

Console Setting—None

Service restart required—Yes

StatsShmSize

Description—This entry is no longer used.

StatsWriteInterval

Description—Interval, in minutes, in which statistics are written to the statistic.sts file

Values—0 - 65535

Default—5

Console Setting—Edit Server Properties page, Logging section, Write interval

Service restart required—No

SystemMemoryLimit

Description—Set by the Double-Take service, each time it is started, to record the amount of available memory.

TargetPaused

Description—Internal setting that indicates if the target machine is paused. Do not modify this setting.

TargetPausedVirtual

Description—Internal setting that indicates which target machines are paused. Do not modify this setting.

TCPBufferSize

Description—Size of the TCP/IP buffer in bytes.

Values—4096-7500000

Default—375000

Console Setting—None

Service restart required—Yes

Notes—The default setting creates a TCP window that will accommodate most environments. In most environments, this value will not need to be adjusted. However, if your Double-Take network has a long end-to-end route and the throughput is not where you would expect it to be, then adjusting this parameter may have beneficial results. This value is the bandwidth delay product, which is calculated using the bandwidth of the network (in bits/second) times the round trip time (in seconds) between the two ends. Use the following recommended settings to improve Double-Take throughput performance.

- 100Mbit LAN—The setting should be around 37500.
- 1Gbit LAN—The setting should be around 375000.
- WAN—The setting should be around 130000.

While the calculations are fairly straight forward, the values that have been suggested are not exact because they depend on round trip time. Some improvements could be gained by adjusting these values either higher or lower. The value suited for your environment can best be determined through trial and error testing.

TempDir

Description—Temporary directory used when replicating Windows 200x encrypted files.

Values—Any valid path

Default—\Program Files\Vision Solutions\Double-Take\Temp

Console Setting—None

Service restart required—No

TGApplyMntPntSecurity

Description—Applies security settings to the volume of a mount point instead of applying them to the directory that the mount point is mounted to.

Values—0 Security will be applied to the directory, 1 Security will be applied to the volume

Default—0

Console Setting—None

Service restart required—Yes

Notes—This setting needs to be applied to the target server.

TGBlockOnConnect

Description—Blocks the target path for all connections, regardless of the source, so that the data cannot be modified

Values—0 Target paths are not blocked, 1 Target paths are blocked

Default—0

Console Setting—None

Service restart required—No

TGCloseDelay

Description—The length of time, in milliseconds, a file is held open on the target

Values—0 - 2000

Default—1000

Console Setting—None

Service restart required—No

Notes—If disk caching on the target is disabled either manually or by default (for example, by default on disks that host Active Directory database files), the target system may be slow during a mirror. If so, descreasing this setting to 100, 10, and 0 will result in incremental improvements, with 0 returning the system performance to normal.

TGDaysToKeepMovedFiles

Description—Specifies the length of time, in days, to keep moved files if TGMoveFilesOnDelete is enabled

Values—any valid integer

Default—0

Console Setting—Edit Server Properties page, Target section, Remove deleted files after this number of days

Service restart required—No

TGDisableAttributeReplication

Description—Specifies whether or not the attributes compression, ACL, and file mask are written to the target during mirroring and replication

Values—0 Enable attribute replication 1 Disable attribute replication

Default—0

Console Setting—None

Service restart required—No

TGExecutionRetryLimit

Description—The number of times an unfinished operation will be retried on the target before it is discarded. If this value is set to zero (0), an operation will never be discarded and will be retried on the target until it is applied.

Values—0 - 65536

Default—0

Console Setting—None

Service restart required—No

TGFileAlloc

Description—Indicates that Double-Take allocates an entire file on the first write of a mirror operation

Values—0 Disabled 1 Enabled

Default—1

Console Setting—None

Service restart required—No

Notes—To help eliminate file fragmentation on the target server, Double-Take should allocate the entire file first. With extremely large files, the file allocation may take a long time. Therefore, you may want to disable the file allocation. If you disable file allocation, you will have more fragmentation on the target disk.

TGMirrorCapacityHigh

Description—Maximum percentage of system memory that can contain mirror data before the target signals the source to pause the sending of mirror operations.

Values—2-75

Default—20

Console Setting—Edit Server Properties page, Target section, Pause mirroring at this level

Service restart required—No

TGMirrorCapacityLow

Description—Minimum percentage of system memory that can contain mirror data before the target signals the source to resume the sending of mirror operations.

Values—1-75

Default—15

Console Setting—Edit Server Properties page, Target section, Resume mirroring at this level

Service restart required—No

Notes—The maximum value forTGMirrorCapacityLow is either 75 or TGMirrorCapacityHigh, which ever is lower.

TGMoveFilesOnDelete

Description—Specifies whether files deleted on the source are actually moved to a different location on the target rather than being deleted on the target

Values—0 Files deleted on the source will be deleted on the target, 1 Files deleted on the source will be moved to a different location on the target

Default—0

Console Setting—Edit Server Properties page, Target section, Moved deleted files to this folder

Service restart required—No

Notes—If this option is enabled, the deleted files will be moved to the location specified in TGMoveFilesPath.

TGMoveFilesPath

Description—Specifies where deleted files on the source are being moved to on the target

Values—any valid path

Default—<null>

Console Setting—Edit Server Properties page, Target section, Moved deleted files to this folder

Service restart required—No

TGMoveFilesSingleDirectory

Description—Specifies if deleted files that will be moved on the target (see **TGMoveFilesOnDelete**) will be moved to a single directory structure

Values—0 Use the same directory structure on the target as the source to store deleted files, 1 Use a single directory structure on the target to store deleted files

Default—0

Console Setting—None

Service restart required—No

TGRetryLocked

Description—Minimum number of seconds to wait before retrying a failed operation on a target

Values—0-65536

Default—3

Console Setting—Edit Server Properties page, Target section, Retry delay for incomplete operations

Service restart required—No

TGThreadCount

Description—This setting is no longer used

TGUnfinishedOpEvent

Description—Specifies whether or not unfinished operations on the target are logged to the Event Viewer

Values—0 Unfinished operation messages are not logged, 1 Unfinished operation messages are logged

Default—1

Console Setting—None

Service restart required—No

TGWriteCache

Description—Specifies whether or not Double-Take uses the intermediate cache

Values—0 Bypass the intermediate cache and write directly to disk, 1 Do not bypass the intermediate cache

Default—0 for full server to ESX appliance jobs, 1 for all other job types

Console Setting—None

Service restart required—No

TGWriteFailureBeforeNotification

Description—Specifies the number of times an operation will be retried on the target before a notification is sent to update the target status

Values—0-1024

Default—10

Console Setting—None

Service restart required—Yes

Notes—If you change the setting to 0, the notification will be disabled. Changing this option will only affect how the target status is displayed. To solve the underlying issue of why the operations are failing will require investigation into the Double-Take log files.

UpdateInterval

Description—This setting is no longer used

UpgradeCode

Description—Used by the Double-Take installation program to maintain the installation settings for an upgrade. Do not modify this entry.

UseChangeJournal

Description—Specifies if the Windows NTFS change journal is used to track file changes. If the source is rebooted, only the files identified in the change journal will be remirrored to the target. This setting helps improve mirror times.

Values—0 Do not track file changes, 1 Track file changes and remirror only changed files on source reboot, 2 Track file changes and remirror only changed files on source reboot and auto-reconnect

Default—1

Console Setting—Edit Server Properties page, Setup section, Mirror only changed files when source reboots

Service restart required—Yes

Notes—The corresponding Console Setting only allows off (0) and on (1) settings. Therefore, if you set UseChangeJournal to 2, the corresponding Console Setting will be disabled. The Console Setting will be reenabled when UseChangeJournal is set to 0 or 1.

UseEventLog

Description—Specifies whether or not messages are logged to the Windows Event Viewer

Values—0 Do not log messages to the Event Viewer, 1 Log messages to the Event Viewer

Default—1

Console Setting—None

Service restart required—No

UseLegacyDrivers

Description—This setting is no longer used

UserIntervention

Description—Specifies whether or not user intervention is required to initiate a failover or cutover

Values—0 User intervention is not required, 1 User intervention is required

Default—1

Console Setting—Set Options page, Failover Options section, Wait for user to initiate failover

Service restart required—No

UseScheduledPause

Description—Used by Double-Take for internal schedule processing. Do not modify this setting.

UseShareFile

Description—Specifies whether to create and use a share file or to use the shares that are currently stored in the target memory

Values—0 Use the shares that are currently stored in the target memory, 1 Create and use a file containing the share information

Default—1

Console Setting—None

Service restart required—No

VerifyLogAppend

Description—Specifies whether the DTVerify.log file will be appended to or overwritten

Values—0 Overwrite, 1 Append

Default—1

Console Setting—Edit Server Properties page, Logging section, Append

Service restart required—No

VerifyLogLimit

Description—Maximum size of the DTVerify.log file in bytes

Values—limited by available hard drive space, up to 4 GB

Default—1048576

Console Setting—Edit Server Properties page, Logging section, Maximum size (under Verification)

Service restart required—No

VerifyLogName

Description—Name of the verification log file

Values—any valid file name

Default—DTVerify.log

Console Setting—Edit Server Properties page, Logging section, File name (under Verification)

Service restart required—No

VerifyRetryInterval

Description—The time, in minutes, between when one verification fails and a retry is scheduled to begin.

Values—any valid number

Default—3

Console Setting—None

Service restart required—No

VerifyRetryLimit

Description—The number of time a verification will be retried.

Values—any valid number

Default—5

Console Setting—None

Service restart required—No

VersionInfo

Description—The version of Double-Take that was installed. Do not modify this entry.

WarningPings

Description—This entry is no longer used.

WatchDogFailureProcessDump

Description—Creates a troubleshooting dump file if the Double-Take driver stops running

Values—0 Do not create a dump file, 1 Create a dump file

Default—0

Console Setting—None

Service restart required—No

WatchDogFailureScript

Description—Specifies the script to run if the Double-Take driver stops running

Values—Any valid path and script file

Default—<null>

Console Setting—None

Service restart required—No

Index

S