


Advanced Active Directory Replication and Topology Management Using Windows PowerShell (Level 200)

 learn.microsoft.com/en-us/windows-server/identity/ad-ds/manage/powershell/advanced-active-directory-replication-and-topology-management-using-windows-powershell--level-200-

In this article

1. [Introduction](#)
2. [See Also](#)

This topic explains the AD DS replication and topology management cmdlets in more detail, and provides additional examples. For an introduction, see [Introduction to Active Directory Replication and Topology Management Using Windows PowerShell \(Level 100\)](#).

1. [Introduction](#)
2. [Replication and Metadata](#)
3. [Get-ADReplicationAttributeMetadata](#)
4. [Get-ADReplicationPartnerMetadata](#)
5. [Get-ADReplicationFailure](#)
6. [Get-ADReplicationQueueOperation](#) and [Get-ADReplicationUpToDatenessVectorTable](#)
7. [Sync-ADObject](#)
8. [Topology](#)

Introduction

The following table lists replication and topology cmdlets added to the Active Directory Windows PowerShell module:

Cmdlet	Explanation
Get-ADReplicationAttributeMetadata	Returns attribute replication metadata for an object
Get-ADReplicationConnection	Returns domain controller connection object details
Get-ADReplicationFailure	Returns the most replication recent failure for a domain controller

Cmdlet	Explanation
Get-ADReplicationPartnerMetadata	Returns replication configuration of a domain controller
Get-ADReplicationQueueOperation	Returns the current replication queue backlog
Get-ADReplicationSite	Returns site information
Get-ADReplicationSiteLink	Returns site link information
Get-ADReplicationSiteLinkBridge	Returns site link bridge information
Get-ADReplicationSubnet	Returns AD subnet information
Get-ADReplicationUpToDatenessVectorTable	Returns the UTD vector for a domain controller
Get-ADTrust	Returns information about an inter-domain or inter-forest trust
New-ADReplicationSite	Creates a new site
New-ADReplicationSiteLink	Creates a new site link
New-ADReplicationSiteLinkBridge	Creates a new site link bridge
New-ADReplicationSubnet	Creates a new AD subnet
Remove-ADReplicationSite	Deletes a site
Remove-ADReplicationSiteLink	Deletes a site link
Remove-ADReplicationSiteLinkBridge	Deletes a site link bridge
Remove-ADReplicationSubnet	Deletes an AD subnet
Set-ADReplicationConnection	Modifies a connection
Set-ADReplicationSite	Modifies a site
Set-ADReplicationSiteLink	Modifies a site link
Set-ADReplicationSiteLinkBridge	Modifies a site link bridge
Set-ADReplicationSubnet	Modifies an AD subnet
Sync-ADObject	Forces replication of a single object

Most of these cmdlets have their basis in Repadmin.exe. Other cmdlets (not listed) handle features like Dynamic Access Control and Group Managed Service Accounts.

For a complete list of all Active Directory Windows PowerShell cmdlets, run:

```
Get-Command -module ActiveDirectory
```

For a complete list of all Active Directory Windows PowerShell cmdlet arguments, reference the help. For example:

```
Get-Help New-ADReplicationSite
```

Use the **Update-Help** cmdlet to download and install help files

Replication and Metadata

Repadmin.exe validates the health and consistency of Active Directory replication. Repadmin.exe offers simple data manipulation options - some arguments support CSV outputs, for example - but automation generally required parsing through text file outputs. The Active Directory module for Windows PowerShell is the first attempt at offering an option that allows real control over the returned data; prior to this, you had to create scripts or use third-party tools.

Additionally, the following cmdlets implement a new parameter set of **Target**, **Scope**, and **EnumerationServer**:

- **Get-ADReplicationFailure**
- **Get-ADReplicationPartnerMetadata**
- **Get-ADReplicationUpToDatenessVectorTable**

The **Target** argument accepts a comma-separated list of strings that identify the target servers, sites, domains, or forests specified by the **Scope** argument. An asterisk (*) is also permissible and means all servers within the specified scope. If no scope is specified, it implies all servers in the current user's forest. The **Scope** argument specifies the latitude of the search. Acceptable values are **Server**, **Site**, **Domain**, and **Forest**. The **EnumerationServer** specifies the server that enumerates the list of domain controllers specified in **Target** and **Scope**. It operates the same as the **Server** argument and requires the specified server run the Active Directory Web Service.

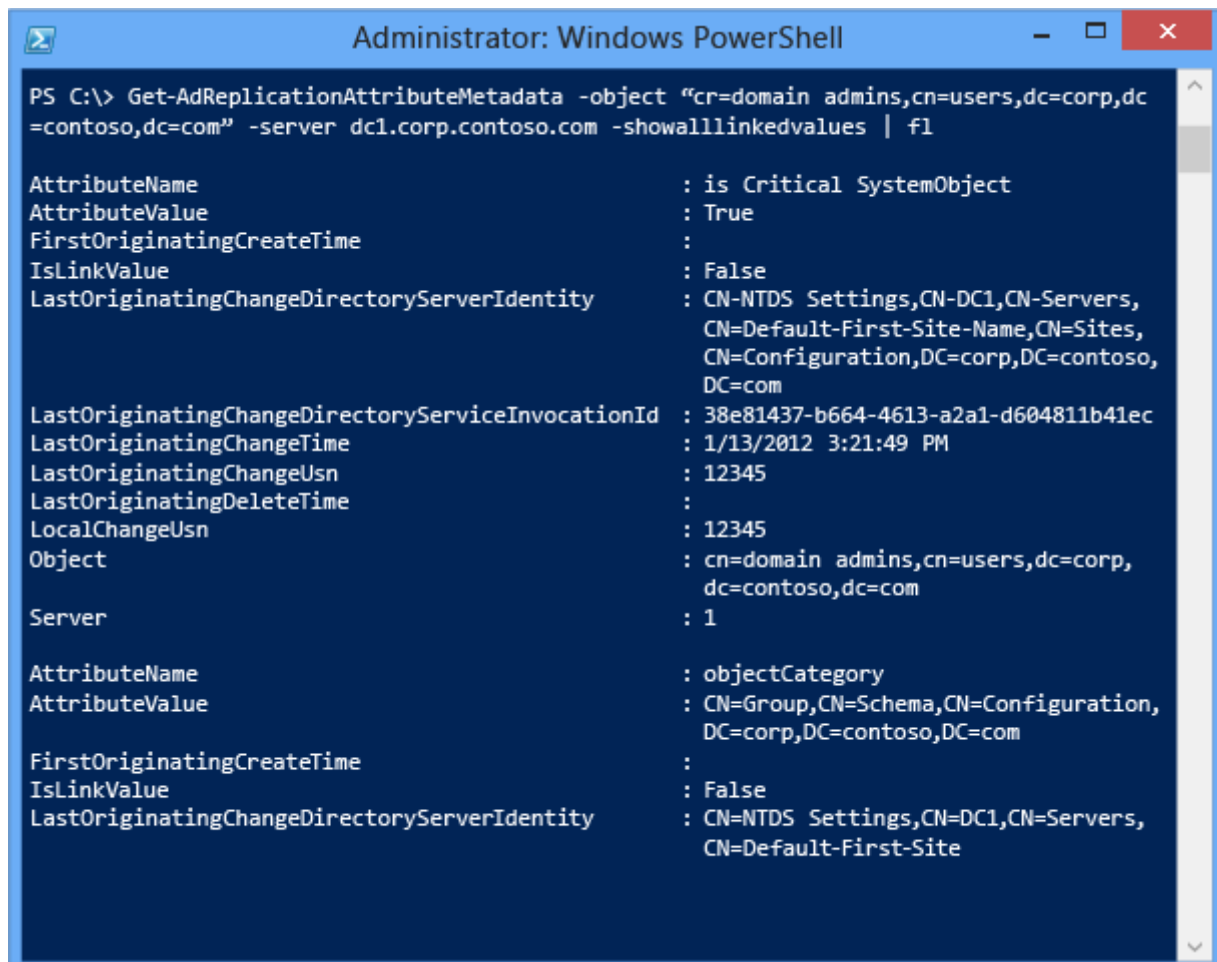
To introduce the cmdlets, here are some sample scenarios showing capabilities impossible to repadmin.exe; armed with these illustrations, the administrative possibilities become obvious. Review the cmdlet help for specific usage requirements.

Get-ADReplicationAttributeMetadata

This cmdlet is similar to **repadmin.exe /showobjmeta**. It enables you to return replication metadata, such as when an attribute changed, the originating domain controller, the version and USN information, and attribute data. This cmdlet is useful for auditing where and when a change occurred.

Unlike Repadmin, Windows PowerShell gives flexible search and output control. For example, you can output the metadata of the Domain Admins object, ordered as a readable list:

```
Get-ADReplicationAttributeMetadata -object "cn=domain  
admins,cn=users,dc=corp,dc=contoso,dc=com" -server dc1.corp.contoso.com -  
showalllinkedvalues | format-list
```



```
Administrator: Windows PowerShell

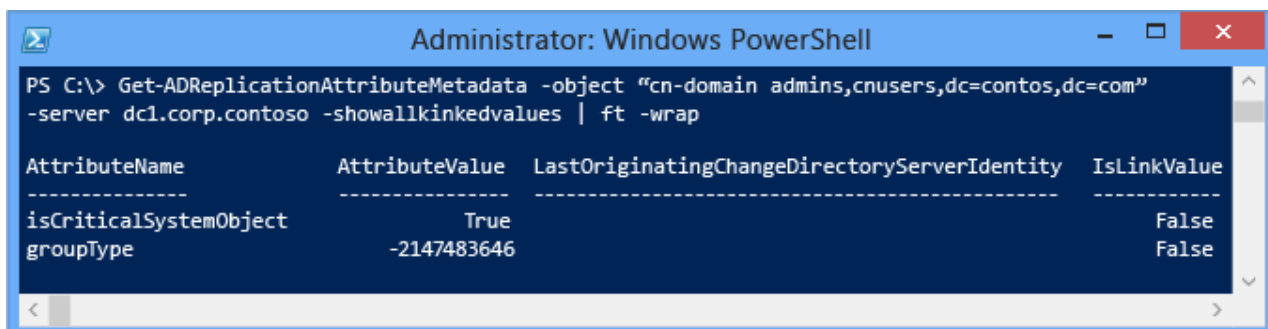
PS C:\> Get-ADReplicationAttributeMetadata -object "cn=domain admins,cn=users,dc=corp,dc=contoso,dc=com" -server dc1.corp.contoso.com -showalllinkedvalues | fl

AttributeName           : is Critical SystemObject
AttributeValue           : True
FirstOriginatingCreateTime :
IsLinkValue              : False
LastOriginatingChangeDirectoryServerIdentity : CN=NTDS Settings,CN=DC1,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=corp,DC=contoso,DC=com
LastOriginatingChangeDirectoryServiceInvocationId : 38e81437-b664-4613-a2a1-d604811b41ec
LastOriginatingChangeTime : 1/13/2012 3:21:49 PM
LastOriginatingChangeUsn : 12345
LastOriginatingDeleteTime :
LocalChangeUsn           : 12345
Object                   : cn=domain admins,cn=users,dc=corp,dc=contoso,dc=com
Server                   : 1

AttributeName           : objectCategory
AttributeValue           : CN=Group,CN=Schema,CN=Configuration,DC=corp,DC=contoso,DC=com
FirstOriginatingCreateTime :
IsLinkValue              : False
LastOriginatingChangeDirectoryServerIdentity : CN=NTDS Settings,CN=DC1,CN=Servers,CN=Default-First-Site
```

Alternatively, you can arrange the data to look like repadmin, in a table:

```
Get-ADReplicationAttributeMetadata -object "cn=domain  
admins,cn=users,dc=corp,dc=contoso,dc=com" -server dc1.corp.contoso.com -  
showalllinkedvalues | format-table -wrap
```



```
Administrator: Windows PowerShell

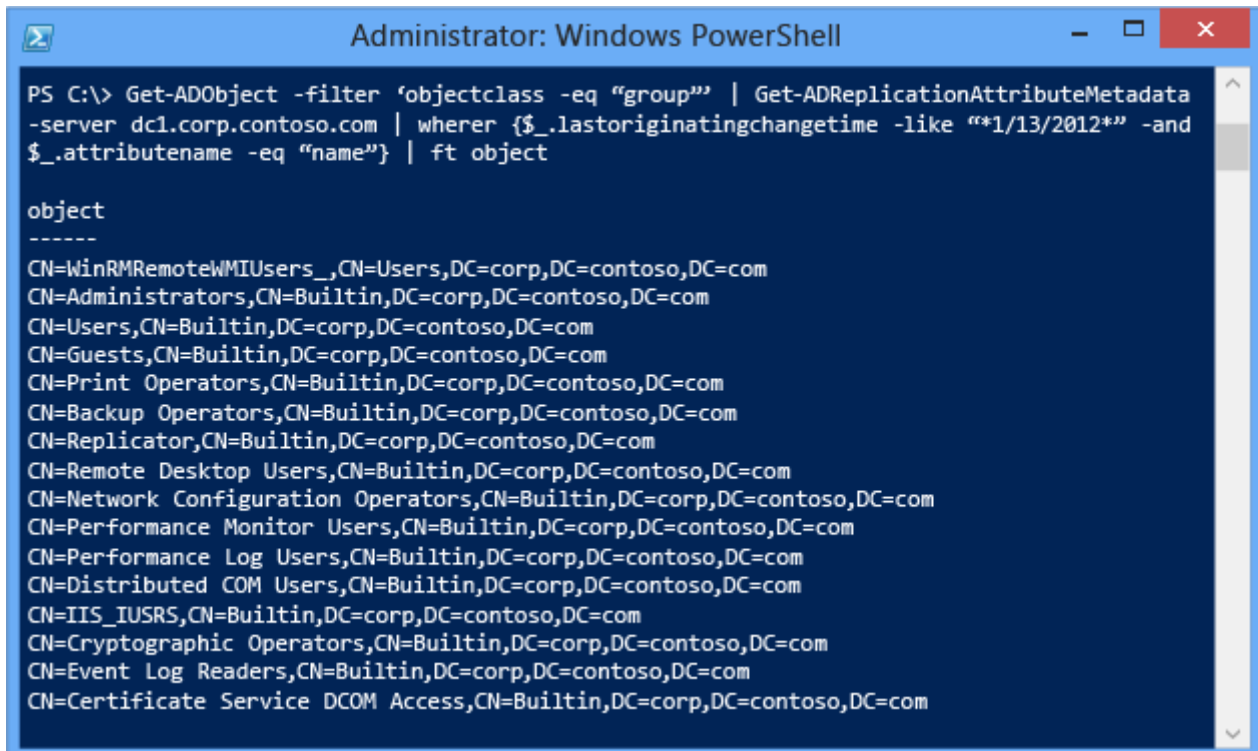
PS C:\> Get-ADReplicationAttributeMetadata -object "cn=domain admins,cnusers,dc=contos,dc=com" -server dc1.corp.contoso -showallkinkedvalues | ft -wrap

AttributeName      AttributeValue  LastOriginatingChangeDirectoryServerIdentity  IsLinkValue
-----
isCriticalSystemObject      True          False
groupType                  -2147483646    False
```

Alternatively, you can get metadata for an entire class of objects, by pipelining the **Get-Adobject** cmdlet with a filter, such as all groups - then combine that with a specific date. The pipeline is a channel used between multiple cmdlets to pass data. To see all groups

modified in some fashion on January 13th, 2012:

```
Get-ADObject -filter 'objectclass -eq "group"' | Get-ADReplicationAttributeMetadata -server dc1.corp.contoso.com | where-object {$_.lastoriginatingchangetime -like "*1/13/2012*" -and $_.attributename -eq "name"} | format-table object
```



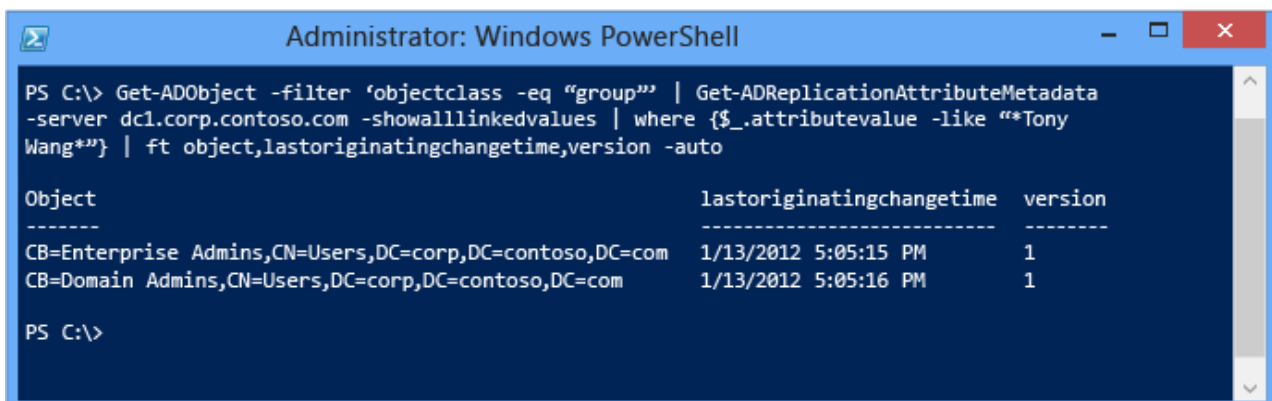
The screenshot shows a Windows PowerShell window titled "Administrator: Windows PowerShell". The command entered is: `PS C:\> Get-ADObject -filter 'objectclass -eq "group"' | Get-ADReplicationAttributeMetadata -server dc1.corp.contoso.com | where {$_.lastoriginatingchangetime -like "*1/13/2012*" -and $_.attributename -eq "name"} | ft object`. The output is a table with one column, "object", listing various built-in groups from the contoso.com domain.

object
CN=WinRMRemoteWMIUsers_,CN=Users,DC=corp,DC=contoso,DC=com
CN=Administrators,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Users,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Guests,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Print Operators,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Backup Operators,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Replicator,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Remote Desktop Users,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Network Configuration Operators,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Performance Monitor Users,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Performance Log Users,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Distributed COM Users,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=IIS_IUSRS,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Cryptographic Operators,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Event Log Readers,CN=Builtin,DC=corp,DC=contoso,DC=com
CN=Certificate Service DCOM Access,CN=Builtin,DC=corp,DC=contoso,DC=com

For more information about more Windows PowerShell operations with pipelines, see [Piping and the Pipeline in Windows PowerShell](#).

Alternatively, to find out every group that has Tony Wang as a member and when the group was last modified:

```
Get-ADObject -filter 'objectclass -eq "group"' | Get-ADReplicationAttributeMetadata -server dc1.corp.contoso.com -showalllinkedvalues | where-object {$_.attributevalue -like "*Tony Wang*"} | format-table object,LastOriginatingChangeTime,version -auto
```

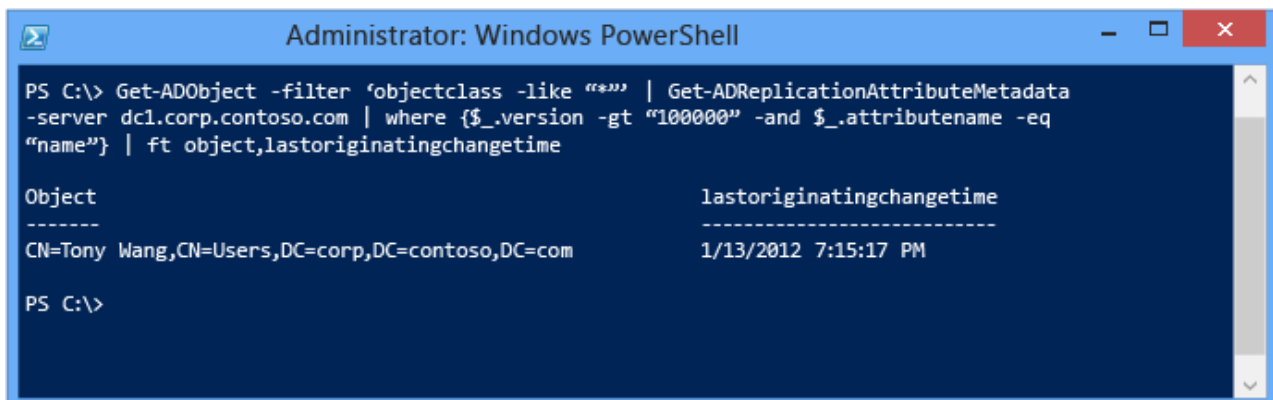


The screenshot shows a Windows PowerShell window titled "Administrator: Windows PowerShell". The command entered is: `PS C:\> Get-ADObject -filter 'objectclass -eq "group"' | Get-ADReplicationAttributeMetadata -server dc1.corp.contoso.com -showalllinkedvalues | where {$_.attributevalue -like "*Tony Wang*"} | ft object,lastoriginatingchangetime,version -auto`. The output is a table with three columns: "Object", "lastoriginatingchangetime", and "version".

Object	lastoriginatingchangetime	version
CB=Enterprise Admins,CN=Users,DC=corp,DC=contoso,DC=com	1/13/2012 5:05:15 PM	1
CB=Domain Admins,CN=Users,DC=corp,DC=contoso,DC=com	1/13/2012 5:05:16 PM	1

Alternatively, to find all objects authoritatively restored using a system state backup in the domain, based on their artificially high version:

```
Get-ADObject -filter 'objectclass -like "*" ' | Get-ADReplicationAttributeMetadata  
-server dc1.corp.contoso.com | where-object {$_.version -gt "100000" -and  
$_.attributename -eq "name"} | format-table object,LastOriginatingChangeTime
```



The screenshot shows a Windows PowerShell window titled "Administrator: Windows PowerShell". The command entered is: `PS C:\> Get-ADObject -filter 'objectclass -like "*" ' | Get-ADReplicationAttributeMetadata -server dc1.corp.contoso.com | where {$_.version -gt "100000" -and $_.attributename -eq "name"} | ft object,LastOriginatingChangeTime`. The output is a table with two columns: "Object" and "LastOriginatingChangeTime". The first row shows the object "CN=Tony Wang,CN=Users,DC=corp,DC=contoso,DC=com" with a change time of "1/13/2012 7:15:17 PM".

Object	LastOriginatingChangeTime
CN=Tony Wang,CN=Users,DC=corp,DC=contoso,DC=com	1/13/2012 7:15:17 PM

Alternatively, send all user metadata to a CSV file for later examination in Microsoft Excel:

```
Get-ADObject -filter 'objectclass -eq "user"' | Get-ADReplicationAttributeMetadata  
-server dc1.corp.contoso.com -showalllinkedvalues | export-csv  
allgroupmetadata.csv
```

Get-ADReplicationPartnerMetadata

This cmdlet returns information about the configuration and state of replication for a domain controller, allowing you to monitor, inventory, or troubleshoot. Unlike Repadmin.exe, using Windows PowerShell means you see only the data that is important to you, in the format you want.

For example, the readable replication state of a single domain controller:

```
Get-ADReplicationPartnerMetadata -target dc1.corp.contoso.com
```

```

Administrator: Windows PowerShell

PS C:\> get-adreplicationpartnermetadata -target dc1.corp.contoso.com -scope server

CompressChanges           : False
ConsecutiveReplicationFailures : 0
DisableScheduledSynch     : False
IgnoreChangeNotifications : False
IntersiteTransport        :
IntersiteTransportGuid    :
IntersiteTransportType    : IP
LastChangeUsn             : 12651
LastReplicationAttempt    : 1/13/2012 5:15:28 PM
LastReplicationResult     : 0
LastReplicationSuccess    : 1/13/2012 5:15:28 PM
Partition                 : DC=corp,DC=contoso,DC=com
PartitionGuid             : 78bd8be9-b0fd-4f82-9544-1f68a2f693da
Partner                   : CN=NTDS Settings,CN=DC2,CN=Servers,
                           CN=Default-First-Site-Name,CN=Sites,
                           CN=Configuration,DC=corp,DC=contoso,
                           DC=com
PartnerGuid               : 753d3f28-0f2b-4fbe-b5ac-7ba9dd019fe1
PartnerInvocationId       : 4dac4bce-7a7c-4e01-8df2-1feb5de11312
PartnerType               : Inbound
ScheduledSync             : True
Server                    : dc1.corp.contoso.com
SyncOnStartup             : True
TwoWaySync                : False
UsnFilter                 : 12651
Writable                  : True

PS C:\>

```

Alternatively, the last time a domain controller replicated inbound and its partners, in a table format:

```
Get-ADReplicationPartnerMetadata -target dc1.corp.contoso.com | format-table
lastreplicationattempt,lastreplicationresult,partner -auto
```

```

Administrator: Windows PowerShell

PS C:\> get-adreplicationpartnermetadata -target dc1.corp.contoso.com | ft lastreplicationattempt,
lastreplicationresult,partner -auto

lastreplicationattempt  lastreplicationresult  partner
-----
1/13/2013 5:34:16 PM    0                      CN=NTDS Settings,CN=DC4,CN=Servers,CN=Defau...
1/13/2013 5:33:58 PM    0                      CN=NTDS Settings,CN=DC4,CN=Servers,CN=Defau...

PS C:\>

```

Alternatively, contact all domain controllers in the forest and display any whose last attempted replication failed for any reason:

```
Get-ADReplicationPartnerMetadata -target * -scope server | where
{$_ .lastreplicationresult -ne "0"} | ft
server,lastreplicationattempt,lastreplicationresult,partner -auto
```



```

Administrator: Windows PowerShell
PS C:\> get-adreplicationpartnermetadata -target * -scope server | where {$_.lastreplicationresult -ne "0"} | ft server,lastreplicationattempt,lastreplicationresult,partner -auto

```

server	lastreplicationattempt	lastreplicationresult	partner
dc1.corp.contoso.com	1/13/2013 5:44:57 PM	1753	CN=NTDS Settings,CN=DC4...
DC2.corp.contoso.com	1/13/2013 5:46:12 PM	1753	CN=NTDS Settings,CN=DC4...

Get-ADReplicationFailure

This cmdlet can be used to returns information about recent errors in replication. It is analogous to **Repadmin.exe /showreplsum**, but again, with much more control thanks to Windows PowerShell.

For example, you can return a domain controller's most recent failures and the partners it failed contacting:

```
Get-ADReplicationFailure dc1.corp.contoso.com
```

```

Administrator: Windows PowerShell
PS C:\> Get-ADReplicationFailure dc1.corp.contoso.com

```

```

FailureCount      : 1
FailureType       : Link
FirstFailureTime  : 1/13/2012 5:34:43 PM
LastError         : 1753
Partner          : CN=NTDS Settings,CD=DC4,CN=Servers,CN=Default-First-Site-Name,CN-Sites,
                  CN-Configuration,DC-corp,DC-contoso,DC-com
PartnerGuid       : 9c3d3a93-f3b1-0463b-893c-d26798ea18b3
Server           : dc1.corp.contoso.com

```

Alternatively, return a table view for all servers in a specific AD logical site, ordered for easier viewing and containing only the most critical data:

```
Get-ADReplicationFailure -scope site -target default-first-site-name | format-table server,firstfailuretime,failurecount,lasterror,partner -auto
```

```

Administrator: Windows PowerShell
PS C:\> get-adreplicationfailure -scope site -target default-first-site-name | ft server,firstfailuretime,failurecount,lasterror,partner -auto

```

server	firstfailuretime	failurecount	lasterror	partner
dc1.corp.contoso...	1/13/2013 5:34:4...	2	1256	CN=NTDS Settings...
DC2.corp.contoso...	1/13/2013 5:44:0...	1	1256	CN=NTDS Settings...
DC3.corp.contoso...	1/13/2013 5:50:5...	1	1253	CN=NTDS Settings...

Get-ADReplicationQueueOperation and Get-ADReplicationUpToDateVectorTable

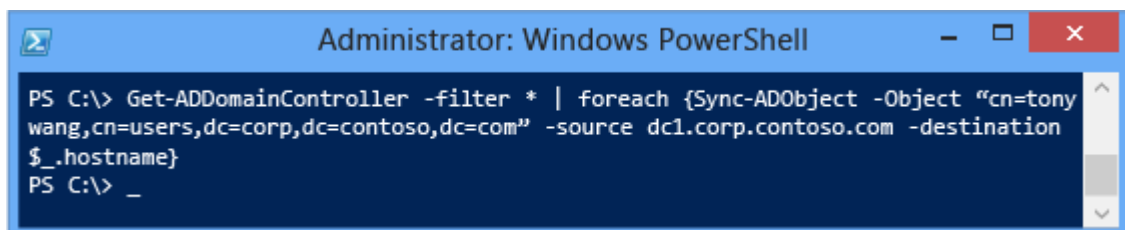
Both of these cmdlets return further aspects of domain controller and whether it's up to date, which includes pending replication and version vector information.

Sync-ADObject

This cmdlet is analogous to running **Repadmin.exe /replsingleobject**. It is very useful when you make changes that require out of band replication, especially to fix an issue.

For example, if someone deleted the CEO's user account and then restored it with the Active Directory Recycle Bin, you probably want it replicated to all domain controllers immediately. You also probably want to do this without forcing replication of all the other object changes made; after all, that is why you have a replication schedule - to avoid overloading WAN links.

```
Get-ADDomainController -filter * | foreach {Sync-ADObject -object "cn=tony wang,cn=users,dc=corp,dc=contoso,dc=com" -source dc1 -destination $_.hostname}
```



Topology

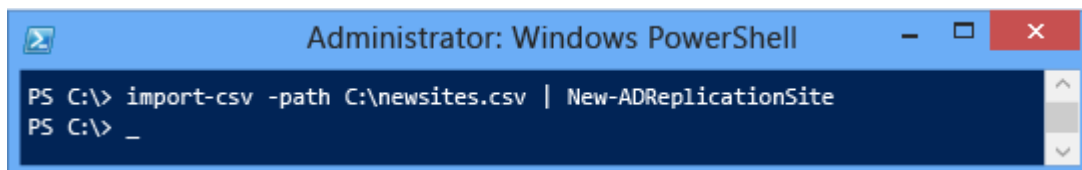
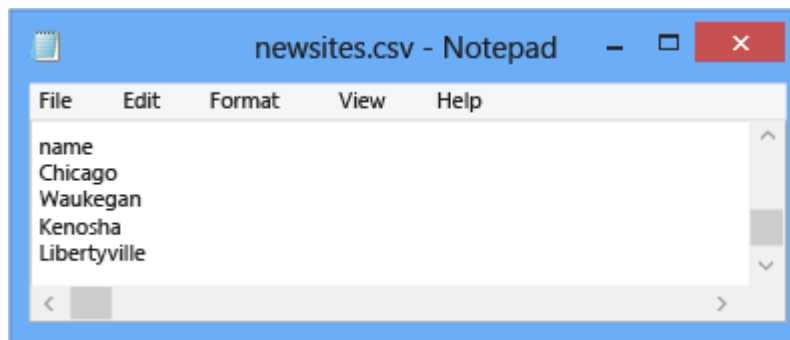
While Repadmin.exe is good at returning information about replication topology like sites, site links, site link bridges, and connections, it does not have a comprehensive set of arguments to make changes. In fact, there has never been scriptable, in-box Windows utility designed specifically for administrators to create and modify AD DS topology. As Active Directory has matured in millions of customer environments, the need to bulk modify Active Directory logical information becomes apparent.

For example, after a rapid expansion of new branch offices, combined with the consolidation of others, you might have a hundred site changes to make based on physical locations, network changes, and new capacity requirements. Rather than using Dssites.msc and Adsiedit.msc to make changes, you can automate. This is especially compelling when you start with a spreadsheet of data provided by your network and facilities teams.

The **Get-Adreplication*** cmdlets return information about replication topology and are useful for pipelining into the **Set-Adreplication*** cmdlets in bulk. **Get** cmdlets do not change data, they only show data or to create Windows PowerShell session objects that can be pipelined to **Set-Adreplication*** cmdlets. The **New** and **Remove** cmdlets are useful for creating or removing Active Directory topology objects.

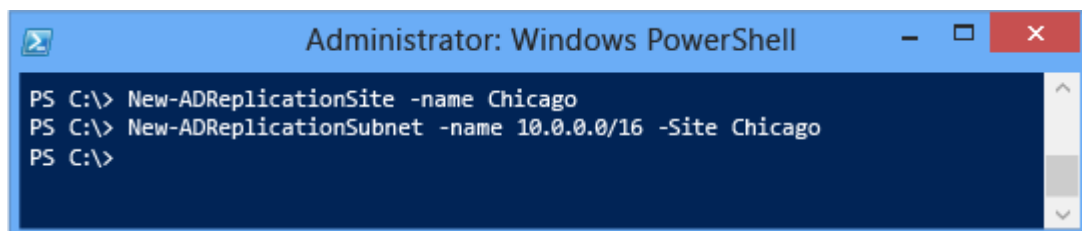
For example, you can create new sites using a CSV file:

```
Import-Csv -path C:\newsites.csv | new-adreplicationsite
```



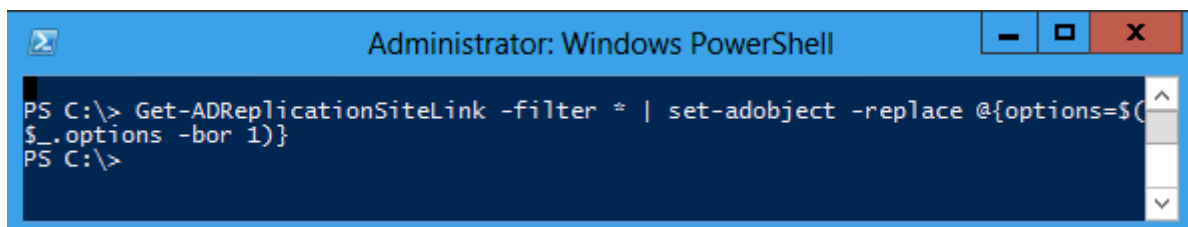
Alternatively, create a new site link between two existing sites with a custom replication interval and site cost:

```
New-ADReplicationSiteLink -name "chicago<-->waukegan" -sitesincluded  
chicago,waukegan -cost 50 -replicationfrequencyinminutes 15
```



Alternatively, find every site in the forest and replace their **Options** attributes with the flag to enable inter-site change notification, in order to replicate at maximum speed with compression:

```
Get-ADReplicationSiteLink -filter * | set-adobject -replace @{options=$($_.options  
-bor 1)}
```

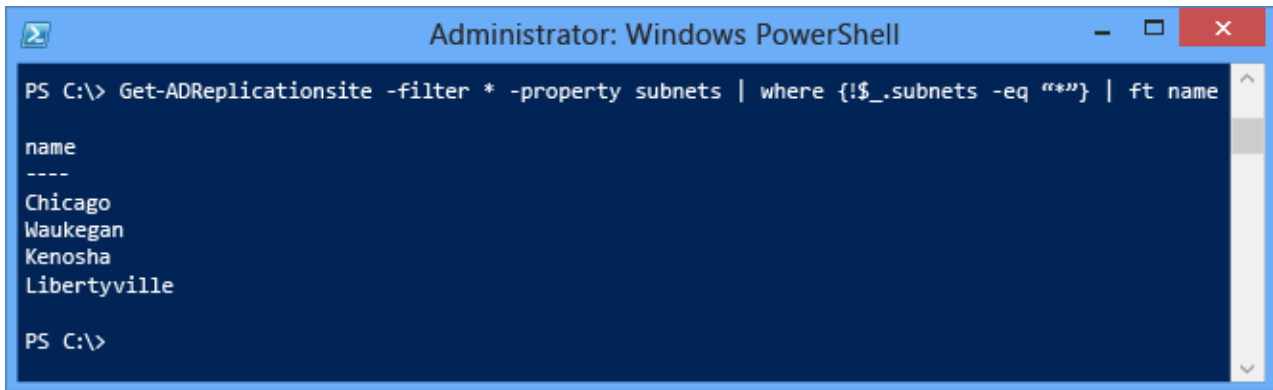


Important

Set **-bor 5** to disable compression on those site links as well.

Alternatively, find all sites missing subnet assignments, in order to reconcile the list with the actual subnets of those locations:

```
Get-ADReplicationSite -filter * -property subnets | where-object {$_.subnets -eq ""} | format-table name
```



The screenshot shows a Windows PowerShell window titled "Administrator: Windows PowerShell". The command entered is `PS C:\> Get-ADReplicationsite -filter * -property subnets | where {$_.subnets -eq ""} | ft name`. The output is a table with one column, "name", and four rows of site names: "Chicago", "Waukegan", "Kenosha", and "Libertyville".

```
PS C:\> Get-ADReplicationsite -filter * -property subnets | where {$_.subnets -eq ""} | ft name

name
----
Chicago
Waukegan
Kenosha
Libertyville

PS C:\>
```

See Also

[Introduction to Active Directory Replication and Topology Management Using Windows PowerShell \(Level 100\)](#)