How to Back Up and Restore Group Policy Objects (GPOs)

blog.netwrix.com/2022/10/18/how-to-backup-and-restore-group-policy-objects

Group Policy objects are critical for managing Windows Server infrastructure. To avoid severe service issues, administrators must configure GPOs carefully and be prepared to revert any changes quickly by backing them up before modifying them.

Handpicked related content:

[Free Guide] Active Directory Group Management Best Practices

Group Policy Refresher

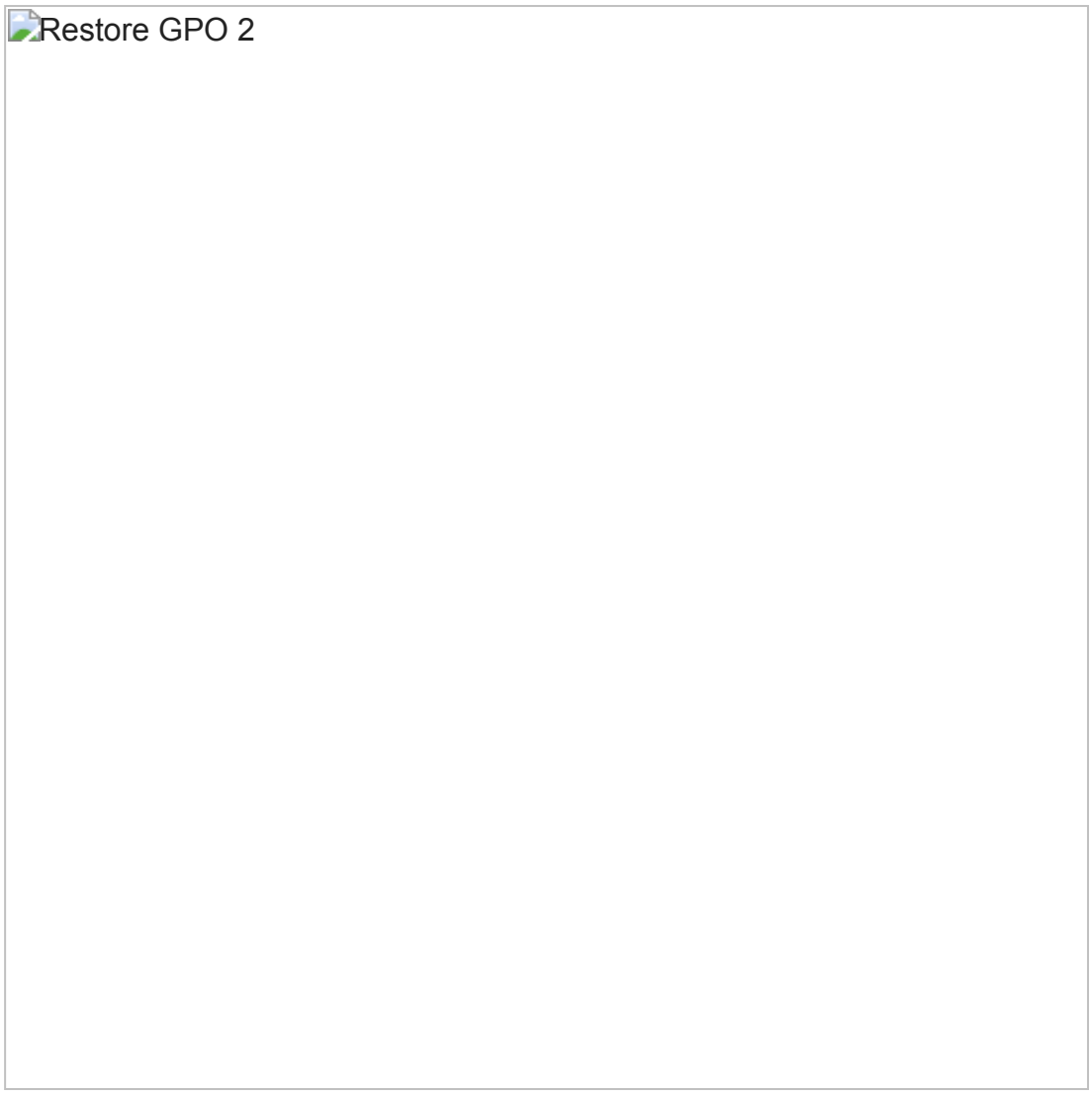
Before we dive into Group Policy object backup and recovery, let's review some important details about how GPOs are created and used.

GPO Components

When a new Group Policy object is created, it is assigned a unique identifier called a GUID. Each GPO has two parts:

> **Group Policy template (GPT)** — The GPT comprises a set of folders in the SYSVOL file share ("C:\Windows\SYSVOL\domain\Policies\{*GUID*}"). These folders are used to store the majority of the content of a Group Policy object, including the templates, settings, scripts and details about MSI packages. The GPT is replicated to every DC in the domain by File Replication Services (FRS) or Distributed File System Replication (DFSR), depending on the version of Windows. In fact, GPOs are effectively domain-specific because SYSVOL is replicated only within a domain.

Group Policy container (GPC) — The GPC is a groupPolicyContainer object located in the domain naming context under CN=System,CN=Policies. This AD object's attributes are used to store referential information related to the GPO. Significantly, this includes the gPCFileSysPath attribute, which contains the path to the GPO's GPT in SYSVOL. Unlike the GPT, the GPC is replicated by Active Directory Domain Services according to the configured replication cost, schedule and interval.



GPO Associations

Once a GPO has been created, it can be associated with one or more Active Directory objects: organizational units (OUs), domains and the sites. This association is not maintained by the GPO but by each of the associated AD objects, in its gPLink attribute. The value of an object's gPLink attribute is a list of the GPC paths of each GPO that the object has been associated with. When a GPO's association to an object is created or deleted, only the value of the affected object's gPLink attribute is modified.

An important remark is that while the replication of SYSVOL effectively makes GPOs domain-specific, the fact that GPOs can be linked to a site object means that GPO-related information is not necessarily confined to a domain. Site objects are stored in the Active Directory Configuration partition, which is replicated to all domain controllers in the forest. This results in the path to a Group Policy object's GPC contained in the gPLink attribute sneaking out of the domain during Active Directory replication.

Group policy processing order

Active Directory applies GPOs in the following order:

1. Local GPOs
2. Site-linked GPOs
3. Domain-linked GPOs
4. OU-linked GPOs (processed beginning from the root, so a GPO linked to a nested OU will take precedence over a GPO linked to its parent OU)

The policy that is applied last “wins” (unless the “Enforce” option is used, which prevents a policy from being overridden by a subsequently applied policy).

User and computer configurations

One last thing to note is that GPOs contain both a Computer Configuration and a User Configuration. These subgroups contain nearly identical sets of Policy settings, but they differ in when the GPO settings are applied.

Computer Configuration settings are applied to computers during boot, and User Configuration settings are applied during logon. This means that options in the Computer Configuration are always enforced against associated computers, while options in the User Configuration are enforced only when an associated user account is logged on to a computer.

If a user logs on to a computer and there is a conflict between a Computer Configuration setting and a User Configuration setting, the Computer Configuration setting will always take precedence.

GPO backup and restore process using PowerShell

With those key concepts fully understood, we can move on to GPO backup and restore.

We’ll begin with the two relevant Group Policy PowerShell cmdlets available as part of Microsoft’s Windows Remote Server Administration Tools:

- **Backup-GPO** — This cmdlet makes it very easy to take a snapshot of all of a domain’s Group Policy objects or a single specified GPO.
- **Restore-GPO** — This cmdlet can restore a Group Policy object to its state as captured in a backup made by the Backup-GPO cmdlet.

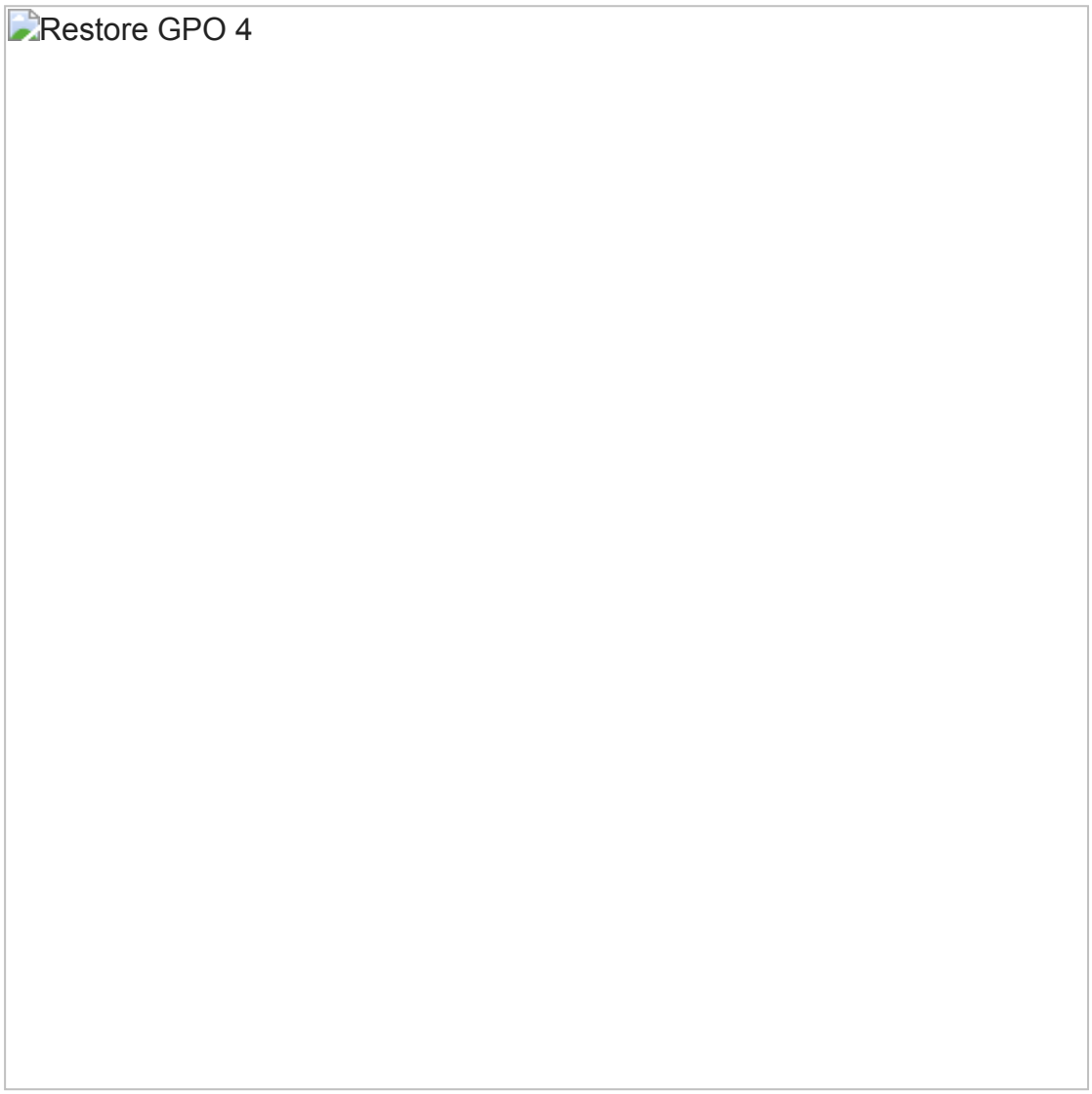
Creating a backup of all GPOs

The following PowerShell script uses the Backup-GPO cmdlet’s -All parameter to create backups of all of the GPOs in the specified domain using a specified DC:

```
$BackupPath = '\HOSTNAMEGPOBackup'
$Domain = 'domain.local'
$DomainController = 'DC.domain.local'

$BackupFolder = New-Item -Path $BackupPath -Name (Get-Date -format
yyyyMMddTHHmmss) -ItemType Directory
Backup-GPO -All -Domain $Domain -Server $DomainController -Path $BackupFolder
```

The output of the Backup-GPO cmdlet consists of a separate subfolder for each GPO's backup information and a manifest.xml file that contains the information necessary to associate each of the subfolders to their respective GPO:



Looking inside one of the subfolders, we find that each backup consists of a folder and three XML files:

The folder contains a copy of the Group Policy object's GPT, and the XML files contain the data from the Group Policy's GPC, information specific to the backup's execution and a report that describes the contents of the GPO.

Backup version control

Repeatedly backing up Group Policy objects to a single location is supported, but each execution of this script creates a unique subfolder for its output. The subfolders are named using backup-specific GUIDs generated during cmdlet execution, which all but eliminates the chance of naming collisions with repeated backups to a single location.

Segregating the output of each execution into unique folders is not strictly necessary, but the behavior of the Restore-GPO cmdlet creates a benefit for doing so. The Restore-GPO cmdlet will allow you to restore all GPOs at once, but it will use the most recent backup of each Group Policy object as identified within the manifest.xml. By separating each set of

backups into its own folder, you ensure that each set of backups gets its own manifest.xml. This allows the restoration of all of the GPOs in any of these backup sets in a single operation.

A downside to using the Backup-GPO cmdlet's -All parameter is that it backs up all of the Group Policy objects in the domain every time it is executed, rather than only the ones that have changed. One way to get around that issue is by using a script like the one below. It forces the Backup-GPO cmdlet to use a file the script creates in the update folder to maintain the timestamp of the script's last runtime. If the file exists, the script backs up any of the domain's GPOs that have been modified since the timestamp in the file. If the file doesn't exist, the script creates the file, sets the timestamp and backs up all of the domain's GPOs.

```
$BackupPath = '\DCGPOBackup'
$Domain = 'domain.local'
$DomainController = 'DC.domain.local'

$BackupPathTracker = "$BackupPathLastBackup.txt"

if((Test-Path "$BackupPathTracker")) {
    $LastBackup = Get-Date (Get-Content -Path "$BackupPathTracker")
    Set-Content -Path "$BackupPathTracker" -Value (Get-Date)
    $GPOs = Get-GPO -All -Domain $Domain -Server $DomainController
    $GPOs | Where-Object { $_.ModificationTime -gt $LastBackup } | ForEach-Object
{
        Backup-GPO -Guid $_.Id -Domain $Domain -Server $DomainController -Path
$BackupPath
    }
} else {
    Set-Content -Path "$BackupPathTracker" -Value (Get-Date)
    Backup-GPO -All -Domain $Domain -Server $DomainController -Path $BackupPath
}
```

While this approach will save space by limiting unnecessary backups, the backups it does make all end up in the same folder, which makes it difficult to restore Group Policy objects to a specific point in time.

Combining the approaches taken in each of the two scripts would solve all of our problems, right?

```
BackupPath = '\HOSTNAMEGPOBackup'
$Domain = 'domain.local'
$DomainController = 'DC.domain.local'

$BackupPathTracker = "$BackupPathLastBackup.txt"
$BackupFolder = New-Item -Path $BackupPath -Name (Get-Date -format
yyyyMMddTHHmmss) -ItemType Directory

if((Test-Path "$BackupPathTracker")) {
    $LastBackup = Get-Content -Path "$BackupPathTracker"
    Set-Content -Path "$BackupPathTracker" -Value (Get-Date)
    $GPOs = Get-GPO -All -Domain $Domain -Server $DomainController
    $GPOs | Where-Object { $_.ModificationTime -gt $LastBackup } | ForEach-Object
{
        Backup-GPO -Guid $_.Id -Domain $Domain -Server $DomainController -Path
$BackupFolder
    }
}
else {
    Set-Content -Path "$BackupPathTracker" -Value (Get-Date)
    Backup-GPO -All -Domain $Domain -Server $DomainController -Path $BackupFolder
}
```

As it turns out, isolating each of the differential backup results in their own folders actually makes everything far worse. This approach also isolates each of the manifest.xml files in their own folders, both effectively eliminating the ability to restore all GPOs at once while also making it incredibly difficult to find the backups associated with any specific Group Policy object.

Restoring GPOs from backup

In practice, these approaches are running into a limitation of the Restore-GPO cmdlet: Restore-GPO can restore a specific Group Policy object from either the most recent backup referenced in a manifest.xml file or a specified backup — but restoring all of a domain's GPOs at once is limited to using the most recent backups in a specified manifest.xml file.

```
# Restore a single GPO from its most recent backup
Restore-GPO -Name 'GpoName' -Path '\HOSTNAMEGPOBackup' -Domain 'domain.local' -
Server 'DC.domain.local'

# Restore a single GPO from a specific backup
Restore-GPO -BackupId 12345678-09ab-cdef-1234-567890abcdef -Path
'\HOSTNAMEGPOBackup' -Domain 'domain.local' -Server 'DC.domain.local'

# Restore all of a domain's GPOs from their most recent backup
Restore-GPO -All -Path '\HOSTNAMEGPOBackup' -Domain 'domain.local' -Server
'DC.domain.local'
```

When a Group Policy object is restored from backup, the version of the GPO is incremented as part of the restoration process in order to force replication to favor the restored copy of the GPO.

How version numbers can get out of sync

The output from the Restore-GPO cmdlet includes two sets of version numbers, one for the Computer Configuration and one for the User Configuration.

The GPT and the GPC are each responsible for maintaining their own version number, with the separate UserVersion and ComputerVersion values being calculated from the User Configuration version and Computer Configuration version numbers, respectively. This is possible as a result of the way the version number is incremented. It is increased by 1 when a GPO's Computer Configuration is modified and by 65536 each time the User Configuration is modified.

All of this is necessary because, as discussed above, a Group Policy object's GPT and GPC are replicated separately by different services, which can result in the version numbers of a GPO's GPT and GPC on any specific domain controller being out of sync — which will prevent it from being processed.

Restoring a deleted GPO

A documented limitation of the Restore-GPO cmdlet is that it cannot be used to recover a Group Policy object that has been deleted, because it will be unable to find the GPC piece of the GPO in Active Directory. Attempting to recover a deleted GPO will result in an error that looks like this:

However, if you are able to recover the GPC first, Restore-GPO can be used to recover the deleted GPO. To do so, we can use the Restore-ADObject cmdlet to fully recover the Active Directory piece of the GPO from the Active Directory Recycle Bin, and then the Restore-GPO cmdlet is able to restore the GPO:

While this process can recover a deleted GPO, it cannot restore the gPLink values that existed prior to the GPO's deletion. This is because those values existed only on the linked objects. The only safe way around this limitation is to leverage external backups of Active Directory that contain the gPLink values.

Using GPMC and AGPM for GPO backup and restore

Group Policy PowerShell cmdlets are not your only option for GPO backup and restore. Microsoft also provides the Group Policy Management Console (GPMC), an MMC snap-in that can be used to back up and restore Group Policy Objects. Like the Backup-GPO cmdlet, it can back up either a single specified GPO or all of a domain's GPOs. Unlike the Restore-GPO cmdlet, it is limited to restoring a single GPO at a time.

GPMC does provide a method to restore deleted GPOs from backup, but it doesn't actually recover the deleted GPO; it actually just creates a new GPO and populates it using the information in the backup.

Another benefit of the GPMC is improved visibility into the contents of the GPO backups, though it remains difficult to compare the settings in a backup to the current settings of the live GPO.

Microsoft's Advanced Group Policy Management (AGPM) tool, which is available as part of the Microsoft Desktop Optimization Pack, extends the GPMC with version control functionality that helps you view and understand the contents of your backups. However, the benefits of AGPM tend to be outweighed by two factors: It doesn't seem to be very well maintained, and it has a reputation for not playing well with others (for example, modifying an AGPM-managed GPO outside of AGPM can result in corruption of the AGPM database). That said, it's not necessarily a bad tool; I just suggest you mess around with it quite a bit it in a lab environment before attempting to deploy it into production.

How can Netwrix help?

Netwrix Recovery for Active Directory provides a unified web interface that enables you to back up both Active Directory objects and Group Policy objects in a single snapshot, search and manage backups, roll back attribute changes to live objects, and even recover deleted GPOs andtheir associated gPLinks.

FAQ

What command can I use to restore a GPO from backup?

The Windows PowerShell cmdlet Restore-GPO restores either a single GPO or all GPOs to the original domain.

How can I import a GPO from a file?

If you have necessary permissions as an administrator in Advanced Group Policy Management (AGPM) and have exported a GPO into a CAB file, then you can import the GPO settings into a new GPO or an existing GPO by following the steps provided here.

Joe Dibley
Security Researcher at Netwrix and member of the Netwrix Security Research Team. Joe is an expert in Active Directory, Windows, and a wide variety of enterprise software platforms and technologies, Joe researches new security risks, complex attack techniques, and associated mitigations and detections.