Delinea

Server Suite

Access Control and Privilege Management Scripting

Version: 2024.x

Publication Date: 7/26/2024

Server Suite Access Control and Privilege Management Scripting

Version: 2024.x, Publication Date: 7/26/2024

Table of Contents

Scripting Access Control and Privilege Management with PowerShell

Note: Subject matter (not formatting) last updated December 2021 (release 2021.1).

Introduction

Overview

This topic discusses access control and privilege management using PowerShell-based command-line programs. This information is intended to help you develop scripts for creating and populating zones and performing other administrative tasks on Windows computers. With scripts, you can automate the administrative tasks you might otherwise perform using the Access Manager console.

Specifically, the topic describes the Delinea authentication and privilege PowerShell-based command set. These PowerShell cmdlets run on Windows computers and can be used to automate access control and privilege management tasks, such as the creation of Delinea zones, rights, and roles. You can also use the cmdlets to perform other administrative tasks. For example, you can write scripts to add UNIX profiles for Active Directory users and groups to Delinea zones, assign UNIX and Windows users and groups to roles, and manage network information through NIS maps.

Intended audience

This topic provides information for Active Directory administrators who want to use PowerShell scripts to install or maintain Delinea software. This document supplements the help provided within the PowerShell environment using the get-help function. Whereas the get-help function describes each cmdlet in detail, this document introduces the access module for Windows PowerShell objects and how you can use PowerShell cmdlets and scripts to perform access control and privilege management tasks.

This topic assumes general knowledge of Microsoft Active Directory, PowerShell scripts and syntax, and the Windows PowerShell modules used to write scripts for Active Directory. You should also understand the structure of Active Directory, including the Active Directory schema your organization is using.

In addition to scripting skills, you should be familiar with Delinea architecture, terms, and concepts, and understand how to perform administrative tasks for authentication and privilege elevation and for the UNIX platforms you support.

Subtopics

This topic is divided into these subtopics:

- **Developing Scripts for Administrative Tasks:** An introduction to access control and privilege management using Windows PowerShell.
- **Installing the PowerShell Access Module:** How to download and install the module as a separate package.
- **Managing Delinea Objects using Windows PowerShell Scripts:** How to use cmdlets to connect to Active Directory and perform access control and privilege management tasks.

- **Objects and Properties:** Lists the objects defined by the authentication and privilege-elevation PowerShell module and the properties of each object.
- **Adding Users in a One-way Trust Environment:** How to add a user in a one-way trust environment using the authentication and privilege-elevation PowerShell module.
- **Using Predefined Scripts to Generate Reports:** Describes predefined report scripts that are included with the authentication and privilege-elevation PowerShell module and how to configure report output files to generate HTML- and PDF-formatted report files.

Compatibility and Limitations

The information in this topic is intended for use with Server Suite, version 5.1.x or later and Server Suite 2017.2 or later. Although intended to be accurate and up to date, interfaces are subject to change without notice and can become incompatible or obsolete when a newer version of the software is released.

In general, APIs attempt to be backward-compatible but are not guaranteed to work with older versions of the software. Because the authentication and privilege elevation cmdlets are subject to change, enhancement, or replacement, the information in this topic can also become incomplete, obsolete, or unsupported in future versions. If you are unsure whether this topic is appropriate for your software version, consult the Delinea Web site or Delinea Support to find out if another, more appropriate, topic is available.

Developing Scripts for Administrative Tasks

This section introduces access control and privilege management using Windows PowerShell. It consists of the following:

- APIs in the form of PowerShell command-line programs, called cmdlets, that are packaged in Dynamic Link Libraries (DLLs).
- A PowerShell help file that includes complete cmdlet reference information and this scripting guide.
- Sample scripts to illustrate administrative tasks.
- Predefined scripts to generate reports.
- Individual help files for each predefined report script.

On Windows computers, you can use the authentication and privilege elevation module for Windows PowerShell to develop your own custom scripts that access, create, or modify Delinea-specific data in Active Directory.

Getting Started with cmdlets for PowerShell

The access module for PowerShell consists of cmdlets that you can use to manage Delinea-specific information in Active Directory. A *cmdlet* is a lightweight command-line program that runs in the Windows PowerShell environment. In most cases, cmdlets perform a basic operation and return a Microsoft .NET Framework object to the next command in the pipeline.

The cmdlets in the Delinea module enable you to access, create, modify, and remove information about Delinea zones, including details for each zone about the defined user, group, and computer profiles; all aspects of the rights, roles, and role assignments; and the available NIS maps and NIS map entries. You can combine cmdlets and use them in scripts to automate administrative tasks, such as user or group profile provisioning or creating rights, roles, and role assignments.

In most cases, you can use cmdlets to manipulate Delinea objects in any type of zone. However, because the implementation of authorization differs greatly in hierarchical zones from authorization in classic zones, the access module for Window PowerShell cmdlets that enable you to create and work with rights, roles, or role assignments are only applicable in hierarchical zones. You should not use the cmdlets for rights, roles, and role assignments in classic zones.

Managing UNIX information from a Windows Computer

You can use the cmdlets to work with information for any Delinea-managed computer and to manage UNIX profiles and access rights. However, you can only run the cmdlets on Windows-based computers that have the Windows PowerShell command-line shell available. If you want to develop scripts that run on UNIX computers, you can use the ADEdit program (adedit). The ADEdit application provides functionality similar to the cmdlets. For detailed information about using ADEdit, see the ADEdit Command Reference and Scripting topic.

Writing Programs in Other Languages

If you want to develop programs or scripts that run on Windows but outside of the Windows PowerShell environment, you can use any language that supports the Component Object Model (COM) interface. The Delinea COM-based interface is available as part of the Delinea Windows Software Development Kit (SDK). The SDK package is a completely separate API that provides reusable objects that you can call in programs written in .NET or COM-enabled languages. You can, therefore, create or modify your own applications to use these objects in VBScript and JScript or in .NET-compliant (such as C#) languages. For more information about using the COM-based API, see the Windows API Programmer's Guide.

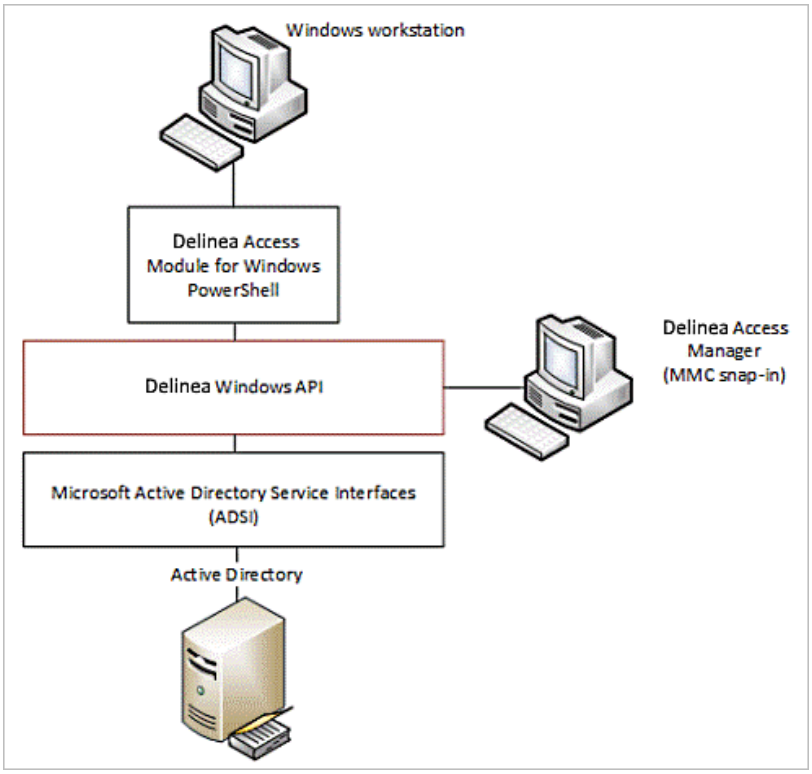
Accessing information stored in Active Directory

The Delinea access module for PowerShell cmdlets connect to Active Directory to access all of the Delinea-specific information stored there. You can, therefore, write PowerShell scripts to automate procedures that you would otherwise have to perform using access manager.

The cmdlets rely on the underlying interfaces provided by Microsoft Active Directory Service Interfaces (ADSI) and the Delinea Windows API. The ADSI layer provides low-level functions that permit applications to read and write data in Active Directory. The cmdlets provide a task and object-based level of abstraction for retrieving and manipulating Delinea-specific information so that you do not need to know the details of how the data is stored or how to use any of the underlying ADSI functions directly.

The following figure illustrates how the Delinea access module for PowerShell provides an abstraction layer between the data stored in Active Directory and your scripting environment.

Figure: PowerShell Abstraction Layer



The Active Directory schema defines how all of the objects and attributes in the database are stored. When you add Delinea objects to the Active Directory database, how that data is stored depends on the Active Directory schema you have installed. The Delinea access module for PowerShell, however, provides a logical view of the data, eliminating the need to know the details of how data is stored in different schemas when performing common administrative tasks. The cmdlets also provide a simple and Delinea-focused method for accessing subject UNIX objects.

Using the cmdlets, you can write scripts that automatically create and manage zones or update user, group, or computer properties. In most cases, the cmdlets enable you to perform exactly the same tasks from the command line that you would otherwise perform interactively using access manager.

Installing the PowerShell Access Module

This section explains how to download and install the module as a separate package. You can install the authentication and privilege elevation module for PowerShell from the Server Suite setup program or as a separate package. This section includes the access control and privilege management cmdlets for Windows PowerShell, sample scripts, and documentation for performing common administrative tasks using PowerShell scripts. This section describes how to install the software if you download it as a separate package or run the package-specific setup program on a Windows computer.

Selecting and Downloading a Standalone Package

The cmdlets that run in Windows PowerShell are defined in DLLs that can be installed on any computer where you install other Windows-based components, such as the Access Manager console. You can also download these

libraries separately, along with sample scripts and documentation, onto computers where access manager is not installed.

Note: You can download the access module for PowerShell as a separate package from the Delinea Download Center under Software Development Kits. However, you must obtain an unlocking code or license key from your Delinea sales representative to access the module.

Running the Setup program

After you have downloaded the compressed file to your computer, you can extract the files and run the setup program to install the access module for PowerShell files.

To use the authentication and privilege elevation module for Windows PowerShell on a Windows Server server-core computer, you must have Windows PowerShell, version 2.0 or later, installed first. Also, install the authentication and privilege elevation module for Windows PowerShell on a Windows Server Core environment in silent mode, due to a user interface limitation. Please check the process exit code to see whether the installation succeeded or failed.

Note: Server core is a minimal installation option that is available when you are deploying Windows Server. Server core includes most but not all server roles. Server Core has a smaller attack surface due to a smaller code base.

To run the standalone setup program:

1. Download the file.
2. Right-click downloaded file and select **Extract All** to extract the compressed files to a folder.
3. Double-click the standalone executable to start the setup program. For example, for the 64-bit version of the file, double click the `CentrifyDC_PowerShell-5.2.0-win64.exe` file.

   Note: Alternatively, you can install from the Microsoft Installer (.msi) file. For example, you might run the following command: `msiexec.exe /i "CentrifyDC_PowerShell-5.2.0-win64.msi" /norestart`.

   The Welcome page appears.
4. Click the **Next** button. The License Agreement page appears.
5. Click to select the **I accept the terms in the License Agreement** check box.
6. Click the **Next** button. The Location page appears.
7. Accept the default location or click **Change** to choose a different one. If you accept the default location, the authentication and privilege elevation cmdlets are in a separate authentication and privilege elevation for Windows PowerShell console. If you want the authentication and privilege elevation cmdlets to be available in the default Windows PowerShell console with other PowerShell modules, select the following location:

   `C:\Windows\System32\WindowsPowerShell\v1.0\Modules\Centrify.DirectControl.PowerShell`
8. Click the **Next** button.
9. Click the **Install** button.
10. Click the **Finish** button to complete the installation.

Importing cmdlets into the Windows PowerShell Console

If you install the authentication and privilege elevation module for Windows PowerShell in the default location, it is a self-contained Windows PowerShell console. If you install the files in the location for system modules so that cmdlets from other modules are available in the same console, you should import the authentication and privilege elevation module into your default Windows PowerShell console.

To import the authentication and privilege elevation module:

1. On the **Start** menu, select **Windows PowerShell** to display a menu extension with a list of tasks.
2. On the tasks menu, select **Import System Modules** to import the authentication and privilege elevation module and open the Windows PowerShell console.
3. Verify the installation and import completed successfully by typing the following command at the PowerShell prompt:

   ```
   get-command -Cdm
   ```

   You should see a listing of the authentication and privilege elevation cmdlets, similar to the following partial list:

```
CommandType   Name                       Definition
-----------   ----                       ----------
 Cmdlet       Add-CdmApplicationRight    Add-CdmApplicationRight -Right ...
 Cmdlet       Add-CdmCommandRight        Add-CdmCommandRight -Right <Cdm...
 Cmdlet       Add-CdmDesktopRight        Add-CdmDesktopRight -Right <Cdm...
 Cmdlet       Add-CdmNetworkAccessRight  Add-CdmNetworkAccessRight -Righ...
 Cmdlet       Add-CdmPamRight            Add-CdmPamRight -Right <CdmPamR...
 Cmdlet       Add-CdmSshRight            Add-CdmSshRight -Right <CdmSshR...
 Cmdlet       Get-CdmApplicationRight    Get-CdmApplicationRight [-Zone ...
 Cmdlet       Get-CdmCommandRight        Get-CdmCommandRight [-Zone <Cdm...
 Cmdlet       Get-CdmComputerRole        Get-CdmComputerRole -Zone <CdmZ...
 Cmdlet       Get-CdmDesktopRight        Get-CdmDesktopRight [-Zone <Cdm...
 Cmdlet       Get-CdmGroupProfile        Get-CdmGroupProfile [-Zone <Cdm...
 ...
```

See Using the Default Windows PowerShell Console for more information..

Managing

This section provides an overview of how to use cmdlets to access and manage authentication and privilege elevation information stored in Active Directory using Windows PowerShell scripts. It provides a summary of the operations you can perform using cmdlets and how to establish a connection to Active Directory. For more examples of how to perform common administrative tasks using the cmdlets, see the samples included with the software.

Using cmdlets to Manage Access

The Delinea access module for PowerShell provides cmdlets that perform operations on objects that correspond to the core elements of Delinea data. Those core elements are:

- Computer role definitions
- Computers
- Groups and group profiles
- NIS network maps and map entries
- Role assignments
- UNIX and Windows rights
- User role definitions
- Users and user profiles
- Zones and zone properties

In most cases, cmdlets can manipulate Delinea information in any type of zone. However, because authorization differs greatly between hierarchical and classic zones, the cmdlets that enable you to work with rights, roles, or role assignments are only applicable in hierarchical zones. You should not use the cmdlets for rights, roles, and role assignments in classic zones. Other than this limitation, you can use the cmdlets to create, access, modify, and remove information associated with any of the core elements of Delinea data for access control and privilege management.

Most of the cmdlets perform one of the following basic operations:

- `Add-CdmXxx` cmdlets add a right to a specified role.
- `Get-CdmXxx` cmdlets get the properties of a specified object.
- `New-CdmXxx` cmdlets create new Delinea objects, such as a new zone or a new role definition.
- `Remove-CdmXxx` cmdlets delete a specified object or remove a right from a specified role.
- `Set-CdmXxx` cmdlets set or change the properties of a specified object.

In addition to these basic operations, there are cmdlets for exporting and importing rights and roles from one zone to another and for establishing connections with Active Directory.

For descriptions of the use and parameters for each cmdlet, use the `get-help` command within the PowerShell console. For example, if you want to see a description and syntax summary for the `New-CdmZone` cmdlet, type the following command in the PowerShell console:

```
get-help New-CdmZone
```

To see detailed information about a cmdlet's parameters and code examples, you can use the -detailed or -full option. For example, type the following command in the PowerShell console:

```
get-help New-CdmZone -detailed
```

Creating and Using a Connection

Because the Delinea access module for PowerShell cmdlets manipulate objects in Active Directory, you must establish a connection with Active Directory before using cmdlets to perform other tasks. To do that, you must specify a target domain or domain controller and the credentials to use when connecting to that domain or domain controller.

Once the credentials are set, all subsequent calls share that information—you do not have to provide the credential or the domain controller for any subsequent calls.

The following example illustrates how to use the administrator account to connect to the finance.acme domain, then add the user joe.doe to the Engineering zone:

```
PS C:\> Set-CdmCredential "finance.acme" "administrator"
PS C:\> Get-CdmCredential
Target         Type   User
------         ----   ----
finance.acme   Forest administrator@finance.acme
PS C:\> $zone = Get-CdmZone -Name "Engineering"
PS C:\> New-CdmUserProfile -Zone $zone -User "joe.doe@finance.acme" -Login "jdoe"
```

In this example, the cmdlets that get the zone and create the user profile use the credential that is cached by the `Set-CdmCredential` command. The `Get-CdmCredential` cmdlet shows what credentials are currently cached.

Managing Connections

You can use the following cmdlets to manage connections to Active Directory by adding, modifying, or using cached credentials or specifying domain-controller-to-domain mappings:

- `Set-CdmCredential` to add or modify a credential in the cache.
- `Get-CdmCredential` to list the credentials currently cached.
- `Set-CdmPreferredServer` to specify a domain controller to use for a domain.
- `Get-CdmPreferredServer` to list all previously defined domain mappings.

Specifying Credentials

You can use the `Set-CdmCredential` cmdlet to specify a credential that you want to cache as a `PSCredential` object. Create the `PSCredential` object using the `Get-Credential` cmdlet. The `Get-Credential` cmdlet prompts users to specify a username and password. You can also pass the username as a parameter to the `Get-Credential` cmdlet to have the cmdlet prompt the user for the password.

Organizing cmdlet Operations in a Sequence

There is no fixed sequence for calling cmdlets. There is, however, a logical sequence to follow to pass data from one cmdlet to another. For example, to get all of the user UNIX profiles in a zone, you must first identify the zone object before you call the `Get-CdmUserProfile` cmdlet, To accomplish this, you could organize the calls in the following sequence:

```
$zone = Get-CdmZone -Name "myZone"
Get-CdmUserProfile -Zone $zone
```

Similarly, to get all of the UNIX user profiles for a computer, you must first identify the computer object:

```
$computer = Get-CdmManagedComputer -Name "myComputer"
Get-CdmUserProfile -Computer $computer
```

In most cases, you can determine from the parameters of a cmdlet whether you need to call another cmdlet first. For example, if you want to add a right to a role, you must have created the role first so it can be specified as a parameter to the `Add-CdmXxx` cmdlet.

For most `Set-CdmXxx` or `Remove-CdmXxx` cmdlets, you must call the corresponding `Get-CdmXxx` or `Add-CdmXxx` cmdlet to obtain the object first. For example, to delete `role1` from `zone1`, you might call the cmdlets as follows:

```
Get-CdmRole -Zone "cn=zone1,cn=Zones,dc=acme,dc=com" -Name "role1" | Remove-CdmRole
```

In this example, the `Get-CdmRole` cmdlet retrieves “role1” from the specified zone and passes it to the `Remove-CdmRole` cmdlet via a PowerShell pipe.

Confirming Licenses

All of the authentication and privilege elevation cmdlets check for a valid license before performing the requested action. The license check succeeds only if there is at least one evaluation, workstation, or server license that has not expired.

If the license check fails, the cmdlet displays an error and stops running. Otherwise, the result is cached. The next time a cmdlet tries to access the same forest, it uses the cached result rather than performing the license check again.

Note: The cache is only effective in one PowerShell console. If another PowerShell console runs a cmdlet accessing the same forest, the cmdlet in that console must perform a separate license check.

Working with Sample Scripts

Introduction

There are several sample scripts included with the software to demonstrate a few common administrative tasks. You can copy and modify these samples to use them in your environment or study them as examples for writing your own custom scripts. The sample scripts include detailed comments about the operations performed to accomplish the following tasks.

Table: Sample Scripts for Administrative Tasks

Sample Script	Demonstrates
`backup.ps1`	How to create a backup copy of a self-contained Delinea zone. This script creates an XML file that contains all computer, user, and group profiles, authorization information, and child zone information for a parent Delinea zone. You cannot use this script to backup SFU zones or child zones.
`CreateZoneAndDelegate.ps1`	How to create a new zone and delegate all zone administrative tasks to a specific trustee.

Sample Script	Demonstrates
`RemoveAllOrphans.ps1`	How to find and delete all user, group, and computer profiles that no longer have a corresponding Active Directory account on all managed computers in each zone.
`RemoveEmptyCompRoles.ps1`	How to find and remove computer roles that have no members. This script is only applicable for hierarchical zones.
`RemoveEmptyZones.ps1`	How to find and remove zones that have no computers, users, or authorization information. This script only removes a zone if it contains no user or group profiles, joined computers, role assignments, computer roles, or child zones. If any of these objects exist for a zone, the zone is not removed. This script is only applicable for hierarchical zones.
`ResetOrphanChildZones.ps1`	How to find child zones that no longer have a parent zone and reset them as independent zones.
`restore.ps1`	How to restore a self-contained Delinea zone from a backup created using the `backup.ps1` sample script.

Running a Sample Script

To run a sample script:

1. Open the Delinea access module for PowerShell.
2. Verify you have permission to execute scripts by running `Get-ExecutionPolicy`. In most cases, the permission to execute scripts is restricted.
3. If necessary, use `Set-ExecutionPolicy` to allow execution. For example:

   ```
   Set-ExecutionPolicy Unrestricted
   ```

   Note: For more about execution policies and the options available, run the `get-help` command.
4. Verify you are in the directory where the scripts are located.
5. Execute the sample script. For example:

   ```
   .\RemoveAllOrphans
   ```

Modifying the Backup and Restore Scripts for Your Needs

If you want to use the sample backup and restore scripts to backup self-contained Delinea zones, you must modify the content of the scripts before executing them. To run a modified sample backup script:

1. Open the `backup.ps1` file in a text editor.
2. Modify the path to the zone you want to back up and the path to the backup file at the start of the sample script. For example:

```
$zoneDn = "CN=Headquarters,CN=Zones,OU=Acme Sales,DC=pistolas,DC=org"
$xmlPath = "C:\Program Files\Centrify\HQ-test.xml"
```

3. Modify the confirmation message at the end of the script to display the path to the backup file. For example:

   ```
   Write-Host "Backup to C:\Program Files\Centrify\HQ-test.xml is done."
   ```

4. Save your changes with a new file name, for example, `HQbackup.ps1`, to keep the sample `backup.ps1` script unchanged.
5. Open the Delinea access module for PowerShell.

Using the Default Windows PowerShell Console

Alternatively, you can use the default Windows PowerShell console. If you choose to use that console, run `import-module` with the path to the access module for PowerShell libraries before performing the above procedure. For example, if you installed the module in the default location, run the following command to import the Delinea access module for PowerShell:

```
import-module “C:\Program Files\Centrify\PowerShell\Centrify.DirectControl.PowerShell.dll”
```

Creating New Zones with the Sample CreateZoneAndDelegate Script

You can use the `CreateZoneAndDelegate.ps1` sample script to automate creating new zones and assigning an Active Directory user or group as the zone administrator. By default, the script delegates all administrative tasks to the user or group you specify. To use the script without modification, simply specify the Active Directory container where you want to create the zone, the zone name, and the user or group designated as the zone administrator.

To create new zone using the sample script:

1. Open the Delinea access module for PowerShell.
2. Verify you are in the directory where the scripts are located.
3. Execute the sample script with the required command line arguments. For example:

   ```
   .\CreateZoneAndDelegate -Container “cn=Zones,ou=Acme Sales,dc=pistolas,dc=org” -ZoneName
   seattle -trustee  frank.smith@pistolas.org
   ```

4. Open Access Manager.
5. Right click **Zones** and select **Open Zone** to search for and select the new zone.
6. If you want to delegate specific administrative tasks, copy the sample script and modify the `Set-CdmDelegation` call to specify a list of tasks. For example:

   ```
   Set-CdmDelegation -Zone $zone -Task "AddUsers","AddGroups” -Trustee $trustee;
   Write-Host "$trustee is delegated the rights to add users and groups.";
   ```

Generating Reports from Predefined Scripts

Most of the predefined reports in access manager report center have a corresponding PowerShell script that you can use to generate reports from the PowerShell console. See Using Predefined Scripts to Generate Reports for

details.

Writing Custom Scripts

Most cmdlets and scripts return information efficiently without any special handling or any noticeable effect on performance. If you plan to write custom scripts that may return large data sets, you should consider ways to improve performance. For example, if you are writing a script that exports a large number of zones or reports on a large number of users, you might want to use the following recommendations as guidelines:

- When testing the performance of the script, use the standard `Measure-Command` cmdlet to accurately measure cmdlet and script performance.

  Note: The `Measure-Command` cmdlet ignores the time it takes to print all of the results returned to the PowerShell console. In many cases, the execution of a script is efficient, but rendering the results in the PowerShell console might make the cmdlet or script performance seem unacceptable.

- Consider how you want to balance memory usage and performance when using the PowerShell pipeline if your cmdlet or script returns large data collections.

  For example, you might use `foreach` in a script instead of using the pipeline to improve performance. Use syntax similar to this:

  ```
  foreach ($cmd in Get-CdmUserProfile -Zone $z) { action_on_each_cmd }
  ```

  Instead of:

  ```
  Get-CdmUserProfile -Zone $z | action_on_each_cmd
  ```

  However, if you choose not to use the pipeline, all of the returned objects stay in memory and might cause an out-of-memory error. Therefore, you should try to maintain balance between the scripts memory usage and performance.

- Cache the data, if possible, by writing the results to a file.

  For example, to add 1000 users to a zone use syntax similar to this to get a zone once:

  ```
  $zone = Get-CdmZone -Dn "cn=QA,cn=Zones,dc=ajax,dc=org"$profile1 = New-CdmUserProfile -Zone $zone -User user1@ajax.org -Uid 10001 ... $profile1000 = New-CdmUserProfile -Zone $zone -User user1000@ajax.org -Uid 11000
  ```

  Instead of using syntax like this, which gets the zone from its distinguished name (DN) for every user:

  ```
  $profile1 = New-CdmUserProfile -Zone "cn=QA,cn=Zones,dc=ajax,dc=org" -User user1@domain.com -Uid 10001 ... $profile1000 = New-CdmUserProfile -Zone "cn=QA,cn=Zones,dc=ajax,dc=org" -User user1000@domain.com -Uid 11000
  ```

- Use `Export-Csv` instead of `Out-File` if possible. The `Export-Csv` cmdlet writes results to a file faster than the `Out-File` cmdlet.
- If you are writing a script that generates a very large data set–for example, reporting information for a global zone–you might want to use the native .NET FileStream function. The FileStream function is the fastest way to write content to a file.

  For example, you might use a code snipper like this:

  ```
  $fs = New-Object IO.FileStream <file>, 'Append','Write','Read'
  ```

```
$fw = New-Object System.IO.StreamWriter $fs
  $zone = Get-CdmZone -Dn "cn=global,cn=Zones,dc=ajax,dc=org"
 foreach ($cz in $zone) {$fw.WriteLine("{0} {1}", $cz.Name, $cz.Type)}
$fw.Close()
$fs.Dispose()
```

Enabling Logging for cmdlets

For performance, logging for cmdlets is disabled by default. To enable logging, you must modify the registry on the computer where you are running the access module for Windows PowerShell.

To enable logging:

1. Run regedit to open the Registry Editor
2. Select the HKEY_CURRENT_USER > Software > Delinea registry key.
3. Right-click, then select New > Key and type CIMS.
4. Select the new CIMS key, right-click, then select New > String Value with the name of LogPath.
5. Specify the path to the log file as the value. For example, set the value to `C: \Temp\Log`.
6. Select the new CIMS key, right-click, then select New > DWORD (32-bit) Value with the name of TraceLevel.
7. Specify the level of detail to write to the log file as the value. The valid settings are:

   0 to disable logging. 1 to only log error messages. 2 to log errors and warning messages. 3 to log errors, warnings, and informational messages. 4 to log all debugging and tracing messages.

   For example, set the value to 4 to enable detailed logging of all messages.

Viewing a Summary of cmdlet Commands

You can use the `get-help` command with different options to get summary about the cmdlets available in the Delinea access module for PowerShell or detailed information about the specific cmdlets you want to use. For example, you can use get-help with the `-full` command-line option to see complete reference information for a specified cmdlet or `get-help -example` to display only the examples for a specified cmdlet.

To see the current list of cmdlets available open the Delinea access module for PowerShell, run the `get-help cdm` command. This command displays a summary of the access module for PowerShell cmdlets similar to the following table (rendered as ASCII characters):

Table: Summary of cmdlet Commands Output by the `get-help cdm` Command

Name	Synopsis
`Add-CdmApplicationRight`	Adds a Windows application right...
`Add-CdmCommandRight`	Adds a UNIX command right to a s...
`Add-CdmDesktopRight`	Adds a Windows desktop right to ...

Name	Synopsis
Add-CdmNetworkAccessRight	Adds a Windows network access ri...
Add-CdmPamRight	Adds a PAM application access ri...
Add-CdmSshRight	Adds an SSH application right to...
Export-CdmData	Exports roles and rights from th...
Get-CdmApplicationRight	Gets an application right from a...
Get-CdmCommandRight	Gets a command right from a zone...
Get-CdmComputerRole	Gets a computer role from a zone.
Get-CdmCredential	Gets user credentials.
Get-CdmDesktopRight	Gets a Windows desktop right fro...
Get-CdmEffectiveGroupProfile	Gets effective group profiles fo...
Get-CdmEffectiveUnixRight	Gets the effective UNIX rights a...
Get-CdmEffectiveUserProfile	Gets effective user profiles for...
Get-CdmEffectiveWindowsRight	Gets the effective Windows right...
Get-CdmGroupProfile	Gets group UNIX profiles.
Get-CdmManagedComputer	Gets zoned or auto-zoned managed...
Get-CdmNetworkAccessRight	Gets a Windows network applicati...
Get-CdmNisMap	Gets NIS maps for the specified ...
Get-CdmNisMapEntry	Gets NIS map entries for the spe...
Get-CdmPamRight	Gets a PAM application access ri...
Get-CdmPreferredServer	Gets domain to server mapping.
Get-CdmRole	Gets roles from a zone.
Get-CdmRoleAssignment	Gets role assignments.
Get-CdmSshRight	Gets an SSH application right fr...

Name	Synopsis
Get-CdmUserProfile	Gets user UNIX profiles.
Get-CdmZone	Gets the zone object.
Import-CdmData	Imports roles and rights into a ...
New-CdmApplicationRight	Creates a new Windows applicatio...
New-CdmCommandRight	Creates a new command right in a...
New-CdmComputerRole	Creates a new computer role in a...
New-CdmDesktopRight	Creates a new Windows desktop ri...
New-CdmGroupProfile	Creates a new UNIX group profile.
New-CdmManagedComputer	Pre-creates a computer or comput...
New-CdmMatchCriteria	Creates a new match criteria for...
New-CdmNetworkAccessRight	Creates a new Windows network ac...
New-CdmNisMap	Creates a new NIS map in a speci...
New-CdmNisMapEntry	Creates a new NIS map entry in a...
New-CdmPamRight	Creates a new PAM application ac...
New-CdmRole	Creates a new role in a zone.
New-CdmRoleAssignment	Creates a new role assignment.
New-CdmUserProfile	Creates a new UNIX user profile.
New-CdmZone	Creates a new zone.
Remove-CdmApplicationRight	Deletes a Windows application ri...
Remove-CdmCommandRight	Deletes a command right or remov...
Remove-CdmComputerRole	Deletes a computer role from a z...
Remove-CdmDesktopRight	Deletes a Windows desktop right ...
Remove-CdmGroupProfile	Deletes a UNIX group profile.

Name	Synopsis
Remove-CdmManagedComputer	Removes a managed computer from ...
Remove-CdmNetworkAccessRight	Deletes a Windows network access...
Remove-CdmNisMap	Deletes a NIS map from a zone.
Remove-CdmNisMapEntry	Deletes a map entry from a NIS map.
Remove-CdmPamRight	Deletes a PAM application access...
Remove-CdmRole	Deletes a role.
Remove-CdmRoleAssignment	Deletes a role assignment from a...
Remove-CdmSshRight	Removes an SSH right from a role.
Remove-CdmUserProfile	Deletes a UNIX user profile.
Remove-CdmZone	Deletes an existing zone.
Set-CdmApplicationRight	Updates an existing Windows appl...
Set-CdmCommandRight	Updates an existing command right.
Set-CdmComputerRole	Updates an existing computer role.
Set-CdmCredential	Adds a user credential.
Set-CdmDelegation	Updates the delegation of admini...
Set-CdmDesktopRight	Updates an existing Windows desk...
Set-CdmGroupProfile	Updates an existing UNIX group p...
Set-CdmNetworkAccessRight	Updates an existing Windows netw...
Set-CdmNisMap	Updates an existing NIS map.
Set-CdmNisMapEntry	Updates an existing NIS map entry.
Set-CdmPamRight	Updates an existing PAM applicat...
Set-CdmPreferredServer	Specifies a preferred server.
Set-CdmRole	Updates an existing role.

Name	Synopsis
Set-CdmRoleAssignment	Updates an existing role assignm...
Set-CdmUserProfile	Updates an existing UNIX user pr...
Set-CdmZone	Updates an existing zone.

Objects and Properties

This section lists the objects defined by the authentication and privilege-elevation PowerShell module and the properties of each object.

This chapter provides an alphabetical listing of the objects and the properties of each object defined in the Access module for PowerShell. Note that not all properties are available as parameters in the PowerShell cmdlets.

CdmAdObject Object

Represents an Active Directory object. The following properties are defined for this object.

Table: CdmAdObject Properties

Property	Type	Description
Class	string	Class of the Active Directory object.
DistinguishedName	string	Distinguished name of the Active Directory object.
Guid	Guid	Globally unique identifier (GUID) of the Active Directory object.
Name	string	Name of the Active Directory object.

CdmAdPrincipal Object

Represents an Active Directory account principal. The following properties are defined for this object.

Table: CdmAdPrincipal Properties

Property	Type	Description
Class	string	Class of the Active Directory object.
DistinguishedName	string	Distinguished name of the Active Directory object.
Guid	Guid	Globally unique identifier (GUID) of the Active Directory object.
Name	string	Name of the Active Directory object.

Property	Type	Description
SamAccountName	string	SAM account name of the Active Directory principal.
Sid	SecurityIdentifier	Security identifier (SID) of the Active Directory principal.

CdmApplicationRight Object

Represents a Windows application access right. This object is only applicable in hierarchical zones. The following properties are defined for this object.

Table: CdmApplicationRight Properties

Property	Type	Description
Description	string	Description of the application right.
IsRequireMfa	Boolean	Indicates whether the application right requires multi-factor authentication.
MatchCriteria	MatchCriteria[]	Filter criteria defined by an array of MatchCriteria objects that identifies the application associated with the application right.
Name	string	Name of the application right.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
Priority	int	Priority of the application right; highest priority prevails.
RequirePassword	Boolean	Indicates whether the application right requires authentication.
RunasSelfGroups	group	The group privileges to add to the user's account when running the application associated with the application right.
RunasUser	user	The user to run the application as.
Zone	zone	Zone where the application right is defined.

CdmCommandRight Object

Represents a UNIX command right. This object is only applicable in hierarchical zones. The following properties are defined for this object.

Table: CdmCommandRight Properties

Property	Type	Description
AddVar	string	Comma separated list of environment variable name-value pairs to add to the final list resulting from KeepVar or DeleteVar property (for instance, "var1=a,var2=b,var3=c").
Authentication	string	The authentication type of the command right: none, user, or runastarget.
DeleteVar	string	Comma separated list of environment variables to remove from default set when command is run.
Description	string	Description of the command right.
Digests	string	Specifies SHA-2 digests so that sudo can verify the binary's checksum (SHA-2) before sudo executes the binary. The supported hash types are sha224, sha256, sha384, and sha512.
DzdoRunAsGroup	string	Comma-separated string of groups allowed to run this command using dzdo (for example, "group1,group2,group3"). The asterisk wild card (*) means any group enabled for the zone can run the command. An empty string ("") means the command cannot run as any group.
DzdoRunAsUser	string	Comma-separated list of users allowed to run this command using dzdo (for example, "user1,user2,user3"). - The asterisk wild card (*) means any user enabled for the zone can run the command. An empty string ("") means the command cannot run as any user.
DzshRunas	string	The user this command will run as under dzsh, '$' means current user.
IsAllowNested	Boolean	True if the command is allowed to start another program or open a new shell.
IsDisablePathTraverse	Boolean	True if the command does not allow navigation up the path hierarchy as an argument.
IsPreserveGroup	Boolean	True to retain the user's group membership while executing a command.
IsRequireMfa	Boolean	Indicates whether the command right requires multi-factor authentication.
KeepVar	string	Comma separated list of environment variables to keep in addition to those in dzdo.env_keep when command is run.

Property	Type	Description
MatchPath	string	The path for matching the command.
Name	string	Name of the command right.
Pattern	string	Command pattern for matching the command.
PatternType	string	The type of pattern—glob or regexp—used to match the command.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
Priority	int	Priority for this command; highest priority prevails.
SELinuxRole	string	Sets the SELinux security context to use the specified role when executing a command using dzdo or dzsh. Applies to command rights on Red Hat Enterprise Linux systems that have SELinux enabled and are joined to a hierarchical zone.
SELinuxType	string	Sets the SELinux security context to use the specified type when executing a command using dzdo or dzsh. Applies to command rights on Red Hat Enterprise Linux systems that have SELinux enabled and are joined to a hierarchical zone.
UMask	string	User file-creation mode mask (umask) value that defines who can execute the command.
Zone	CdmZone	Zone of the command right.

CdmComputer Object

Represents an Active Directory computer object. The following properties are defined for this object.

Table: CdmComputer Properties

Property	Type	Description
Class	string	Class of the Active Directory object.
DistinguishedName	string	Distinguished name of the Active Directory object.
DNSHostName	string	DNS host name of the Active Directory computer.
Enabled	Boolean	True if the Active Directory computer is enabled.
Guid	Guid	GUID of the Active Directory object.
Name	string	Name of the Active Directory object.

Property	Type	Description
SamAccountName	string	SAM account name of the Active Directory principal.
Sid	SecurityIdentifier	SID of the Active Directory principal.
UserPrincipalName	string	User principal name of the Active Directory computer.

CdmComputerRole Object

Represents a Delinea computer role. This object is only applicable in hierarchical zones. The following properties are defined for this object.

Table: CdmComputerRole Properties

Property	Type	Description
CustomAttributes	string	Custom text strings for the computer role.
Description	string	Description of the computer role.
Group	CdmGroup	Computer group associated with this computer role.
Name	string	Name of the computer role.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
Zone	CdmZone	Zone that contains the computer role.

CdmDesktopRight Object

Represents a Windows desktop access right. This object is only applicable in hierarchical zones. The following properties are defined for this object.

Table: CdmDesktopRight Properties

Property	Type	Description
Description	string	Description of the desktop right.
IsRequireMfa	Boolean	Indicates whether the desktop right requires multi-factor authentication.
Name	string	Name of the desktop right.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
Priority	int	Priority of the desktop right; highest priority prevails.

Property	Type	Description
RequirePassword	Boolean	True if the desktop right requires a password.
RunasSelfGroups	CdmGroup[]	Groups whose privileges are added to the user account running the desktop.
RunasUser	CdmUser	User to run the desktop as.
Zone	CdmZone	Zone of the desktop right.

CdmEffectiveUnixRights Object

Represents the UNIX rights assigned to a user that are in effect on a Linux or UNIX computer in a zone. The following properties are defined for this object.

Table: CdmEffectiveUnixRights Properties

Property	Type	Description
AuditLevel	string	Effective auditing level.
CommandRights	CdmEffectiveCommandRight []	The array of effective command rights assigned to the user.
Computer	CdmManagedComputer	The computer in which the roles and rights are effective.
HasRescueRight	Boolean	True if the user has the rescue right.
PamRights	CdmEffectivePamRight[]	The array of effective PAM rights assigned to the user.
Profiles	CdmEffectiveUserProfile[]	Effective UNIX profiles for the Active Directory user in the computer.
Roles	CdmEffectiveRole[]	The array of effective roles assigned to the user.
SshRights	CdmEffectiveSshRight[]	The array of effective SSH rights assigned to the user.
UnixSystemRights	string[]	Effective UNIX system rights.
User	CdmUser	Active Directory user assigned to the role.

CdmEffectiveWindowsRights Object

Represents the Windows rights assigned to a user that are in effect on a Windows computer in a zone. The following properties are defined for this object.

Table: CdmEffectiveWindowsRights Properties

Property	Type	Description
AuditLevel	string	Effective auditing level.
ApplicantionRights	CdmEffectiveApplicationRight	The array of effective application rights assigned to the user.
Computer	CdmManagedComputer	The computer in which the roles and rights are effective.
DesktopRights	CdmEffectiveDesktopRight	The array of effective desktop rights assigned to the user.
HasRescueRight	Boolean	True if the user has the rescue right.
NetworkRights	CdmEffectiveNetworkRigh	The array of effective network access rights assigned to the user.
Roles	CdmEffectiveRole	The array of effective roles assigned to the user.
WindowsSystemRights	string[]	Effective Windows system rights.
User	CdmUser	Active Directory user assigned to the role.

CdmGroup Object

Represents an Active Directory group. The following properties are defined for this object.

Table: CdmGroup Properties

Property	Type	Description
Class	string	Class of the Active Directory object.
DistinguishedName	string	Distinguished name of the Active Directory object.
GroupCategory	ADGroupCategory	Category of the Active Directory group.
GroupScope	ADGroupScope	Scope of the Active Directory group.
Guid	Guid	GUID of the Active Directory object.
Name	string	Name of the Active Directory object.
SamAccountName	string	SAM account name of the Active Directory principal.
Sid	SecurityIdentifier	SID of the Active Directory principal.

CdmGroupProfile Object

Represents a UNIX group profile. The following properties are defined for this object.

Table: CdmGroupProfile Properties

Property	Type	Description
Computer	CdmManagedComputer	Computer that contains the profile.
Gid	long	GID of the group profile.
Group	CdmGroup	Active Directory group of the group profile.
IsHierarchical	Boolean	True if the group profile is in a hierarchical zone.
IsMembershipRequired	Boolean	True if users are required to be a member of this group.
IsOrphan	Boolean	True if the group profile is an orphan profile, that is, it has no corresponding Active Directory group.
IsSfu	Boolean	True if the group profile is a SFU profile.
Name	string	Name of the group profile.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
Zone	CdmZone	Zone that contains the profile.

CdmLocalGroupProfile Object

Represents a local UNIX group profile. The following properties are defined for this object.

Table: CdmLocalGroupProfile Properties

Property	Type	Description
CanonicalName	string	Canonical name of the local group profile.
Computer	CdmManagedComputer	Computer where the local group profile is defined.
Domain	string	Domain of the local group profile.
Gid	long	GID of the group profile.
Members	string[]	Members of the local group profile.

Property	Type	Description
Name	string	Name of the group profile.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
State	enum	State of the local group profile. The valid values are: Enable, Remove. and Inherit The default state is Inherit.
Zone	CdmZone	Zone that contains the profile.

CdmLocalUserProfile Object

Represents a local UNIX user profile. The following properties are defined for this object.

Table: CdmLocalUserProfile Properties

Property	Type	Description
CanonicalName	string	Canonical name of the local user profile.
Computer	CdmManagedComputer	Computer where the local user profile is defined.
Domain	string	Domain of the local user profile.
Gecos	string	GECOS field of the local user profile.
HomeDir	string	Home directory of the user associated with the local profile.
Name	string	Name of the user associated with the local profile.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
PrimaryGroupId	long	Primary group ID of the user associated with the local profile.
Shell	string	Default shell of the user associated with the local profile.
State	enum	State of the local user profile. The valid values are: Enable, Remove, and Inherit The default state is Inherit.
Uid	long	Numeric user identifier (UID) of the user associated with the local profile.
Zone	CdmZone	Zone where the local user profile is defined.

CdmLocalWindowsGroup Object

Represents a local Windows group account. The following properties are defined for this object.

Table: CdmLocalWindowsGroup Properties

Property	Type	Description
CanonicalName	string	Canonical name of the local group in Active Directory.
Computer	CdmManagedComputer	Computer where the local group is defined.
Description	string	Description for the local group.
Domain	string	Domain of the local group in Active Directory.
Members	string[]	Members of the local group .
Name	string	Name of the local group .
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
State	LocalWindowsGroupState enum	State of the local group . The valid values are: Enable, Remove, and Inherit The default state is Inherit.
Zone	CdmZone	The zone where the local group is defined.

CdmLocalWindowsUser Object

Represents a local Windows user account. The following properties are defined for this object.

Table: CdmLocalWindowsUser Properties

Property	Type	Description
CanonicalName	string	Canonical name of the local user account in Active Directory.
Computer	CdmManagedComputer	Computer where the local user is defined.
Description	string	Description for the local user.
Domain	string	Domain of the local user account in Active Directory.
FullName	string	Full name of the local user.
Name	string	Name of the local user.

Property	Type	Description
PasswordOptions	LocalWindowsUserPassword Option enum	Password options of the local user. Possible values are: None, Inherit, UserMustChangePasswordAtNextLogon, UserCannotChangePassword, PasswordNeverExpires. It can be a combination of UserMustChangePasswordAtNextLogon and PasswordNeverExpires, UserCannotChangePassword and PasswordNeveExpires.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
State	LocalWindowsUserState enum	State of the local user. he valid values are: Enable, Remove, and Inherit The default state is Inherit.
Zone	CdmZone	The zone where the local user is defined.

CdmManagedComputer Object

Represents a computer managed by authentication and privilege elevation. The following properties are defined for this object.

Table: CdmManagedComputer Properties

Property	Type	Description
AgentVersion	string	Version number of the Delinea Agent installed on the managed computer.
Computer	CdmComputer	Corresponding Active Directory computer account.
ComputerZonePath	string	Path to the computer zone.
IsComputerZoneOnly	Boolean	True if the managed computer has a computer zone only (that is, the computer is not joined to a zone).
IsExpressMode	Boolean	True if the managed computer is in Express (unlicensed) mode.
IsHierarchical	Boolean	True if the managed computer is joined to a hierarchical zone.
IsOrphan	Boolean	True if the managed computer is an orphan profile, that is, it has no corresponding Active Directory computer object.
IsWindows	Boolean	True if the managed computer is a Windows computer.

Property	Type	Description
IsWorkstationMode	Boolean	True if the managed computer is joined to Auto Zone in Workstation mode.
IsJoinedToZone	Boolean	True if the managed computer is joined to a zone.
LicenseType	string	Type of license being used. This property is Server if the managed computer is a Windows or UNIX server or Workstation if the managed computer is not used as a server.
Name	string	Name of the managed computer.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
ScpPath	string	Path to the service connection point for the managed computer.
Zone	CdmZone	Zone of the managed computer.

CdmMatchCriteria Object

Represents an application right match criteria object defined using the application rights match criteria filters. The following properties are defined for this object.

Table: CdmMatchCriteria Properties

Property	Type	Description
Argument	string	The argument for the application.
CompanyName	string	All or part of the company name associated with the application.
CompanyNameMatchOption	string	Specifies whether the company name string should be an exact match (is) or a partial match (contains).
Description	string	The description for the application criteria.
FileDescription	string	All or part of the file description for the application.
FileDescriptionMatchOption	string	Specifies whether the file description string should be an exact match (is) or a partial match (contains).
FileHash	string	The file hash for an application.
FileName	string	The file name for an application.
FileType	string	The file type for an application.

Property	Type	Description
FileVersion	string	All or part of the file version information for an application.
FileVersionMatchOption	string	Specifies whether the file version string should be an exact match (equal), an earlier or equal version (earlier or equal), or a later or equal version (later or equal).
IsArgumentCaseSensitive	Boolean	True if the argument specified is case sensitive.
IsArgumentExactMatch	Boolean	True if the argument must be matched exactly as specified.
IsRequireAdministrator	Boolean	True if the application requires administrator privileges to execute.
LocalOwner	string	The local owner for the application.
LocalOwnerType	string	The local owner type for the application.
OwnerSid	string	The owner security identifier (SID) for the application.
Path	string	The path to an application.
ProductName	string	All or part of the product name associated with the application.
ProductNameMatchOption	string	Specifies whether the product name string should be an exact match (is) or a partial match (contains).
ProductVersion	string	All or part of the product version information for an application.
ProductVersionMatchOption	string	Specifies whether the product version string should be an exact match (equal), an earlier or equal version (earlier or equal), or a later or equal version (later or equal).
Publisher	string	The publisher for an application.
PublisherMatchOption	string	Specifies whether the publisher string should be an exact match (is), a partial match (contains), start with, or end with the specified string.
SerialNumber	string	The serial number for an application.
SerialNumberMatchOption	string	Specifies whether the serial number string should be an exact match (is), a partial match (contains), start with, or end with the specified string.

CdmNetworkRight Object

Represents a Windows network access right. This object is only applicable in hierarchical zones. The following properties are defined for this object.

Table: CdmNetworkRight Properties

Property	Type	Description
Description	string	Description of the network right.
IsRequireMfa	Boolean	Indicates whether the network access right requires multi-factor authentication.
Name	string	Name of the network right.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
Priority	int	Priority of the network right; highest priority prevails.
RequirePassword	Boolean	True if the network right requires a password.
RunasSelfGroups	CdmGroup[]	Groups whose privileges are added to the user account accessing the network.
RunasUser	CdmUser	Run-as user of the network right.
Zone	CdmZone	Zone of the network right.

CdmPamRight Object

Represents a PAM application access right. This object is only applicable in hierarchical zones. The following properties are defined for this object.

Table: CdmPamRight Properties

Property	Type	Description
Application	string	PAM application for this right.
Description	string	Description of the PAM access right.
Name	string	Name of the PAM access right.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
Zone	CdmZone	Zone of the PAM access right.

CdmRole Object

Represents a authentication and privilege elevation role. This object is only applicable in hierarchical zones. The following properties are defined for this object.

Table: CdmRole Properties

Property	Type	Description
AllowLocalUser	Boolean	True if the role can be assigned to a local user.
AuditLevel	string	Audit setting for this role.
CustomAttributes	string	Custom text strings for the role.
Description	string	Description of the role.
HasRescueRight	Boolean	True if users assigned to this role can log on when problems with authentication, authorization or auditing services prevent log on access.
HasDzdoRescueRight	Boolean	True if this role allows users to run Dzdo when problems with authentication, authorization or auditing services prevent Dzdo operation.
Name	string	Name of the role.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
RequireMfa	Boolean	True if the role requires multi-factor authentication.
TimeBox	Hashtable	Active time of the role.
UnixSystemRights	string[]	UNIX system rights granted to the role.
WindowsSystemRights	string[]	Windows system rights granted to the role.
Zone	CdmZone	Containing zone.

CdmRoleAssignment Object

Represents a authentication and privilege elevation role assignment. This object is only applicable in hierarchical zones. The following properties are defined for this object.

Table: CdmRoleAssignment Properties

Property	Type	Description
AdTrustee	CdmAdPrincipal	The trustee, if it is an Active Directory account.
Computer	CdmManagedComputer	Containing computer.
ComputerRole	CdmComputerRole	Containing computer role.
CustomAttributes	string	Custom text strings for the role assignment.
Description	string	The role assignment description.
EndTime	DateTime	The ending date and time for the role assignment.
IsNeverExpire	Boolean	True if the role assignment never expires.
IsRoleOrphaned	Boolean	True if the role is missing or invalid.
IsStartImmediately	Boolean	True if the role assignment starts immediately.
IsTrusteeOrphaned	Boolean	True if the trustee is missing or invalid.
LocalTrustee	string	The trustee, if it is a local account.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
Role	CdmRole	Assigned role.
StartTime	DateTime	The starting date and time for the role assignment.
TrusteeType	string	Type of trustee.
Zone	CdmZone	Containing zone.

CdmSshRight Object

Represents an SSH application access right. This object is only applicable in hierarchical zones. The following properties are defined for this object.

Table: CdmSshRight Properties

Property	Type	Description
Application	string	Secure shell application for this right.
Description	string	Description of the SSH right.

Property	Type	Description
Name	string	Name of the SSH right.
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
Zone	CdmZone	Zone of the SSH right.

CdmUser Object

Represents an Active Directory user. The following properties are defined for this object.

Table: CdmUser Properties

Property	Type	Description
Class	string	Class of the Active Directory object.
DistinguishedName	string	Distinguished name of the Active Directory object.
Enabled	Boolean	True if the Active Directory user is enabled.
GivenName	string	Given name of the Active Directory user.
Guid	Guid	GUID of the Active Directory object.
IsAllADUser	Boolean	True if the user is an Active Directory domain user account.
Name	string	Name of the Active Directory object.
SamAccountName	string	SAM account name of the Active Directory principal.
Sid	SecurityIdentifier	SID of the Active Directory principal
Surname	string	Surname of the Active Directory user.
UserPrincipalName	string	User principal name of the Active Directory user.

CdmUserProfile Object

Represents a UNIX user profile. The following properties are defined for this object.

Table: CdmUserProfile Properties

Property	Type	Description
Computer	CdmManagedComputer	Containing computer.

Property	Type	Description
ExtendedAttributes	string	AIX extended attributes of the user profile
Gecos	string	GECOS field of the user profile
HomeDirectory	string	Home directory of the user associated with the profile
IsHierarchical	Boolean	True if the user profile is in a hierarchical zone
IsOrphan	Boolean	True if the user profile is an orphan profile, that is, it has no corresponding Active Directory user.
IsSecondary	Boolean	True if the user profile is a secondary profile.
IsSfu	Boolean	True if the user profile is an SFU profile.
IsUseAutoPrivateGroup	Boolean	True if the user private group is to be used as the primary group.
Name	string	Name of the user associated with the profile
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
PrimaryGroupId	long	Primary group ID of the user associated with the profile
Shell	string	Default shell of the user associated with the profile
Uid	long	UID of the user associated with the profile
UnixEnabled	Boolean	True if the user profile is enabled for a classic zone. This property is not applicable in hierarchical zones.
User	CdmUser	Active Directory user for whom this is the user profile
Zone	CdmZone	Containing zone

CdmZone Object

Represents a Delinea zone. The following properties are defined for this object.

Table: CdmZone Properties

Property	Type	Description
AgentlessPasswordAttribute	string	Attribute in which to store the password hash for agentless client.

Property	Type	Description
AvailableShells	string[]	Array of available shells that can be used as the default shell for zone users.
CanonicalName	string	Canonical name of the zone.
CloudInstance	String	Cloud instance URL to which the zone connects.
DefaultGecos	string	Default GECOS field for zone users.
DefaultGid	long	Default GID value to use for zone groups.
DefaultGroupName	string	Default group name to use for zone groups.
DefaultHomeDirectory	string	Default home directory for zone users.
DefaultPrimaryGroup	string	Default primary group to use for zone users.
DefaultShell	string	Default shell for zone users.
DefaultUid	long	Default UID value to use for zone users.
DefaultUserName	string	Default user name to use for zone users.
DefaultValueZone	CdmZone	Zone to use as the source for default values in a selected zone.
Description	string	Description of the zone.
DistinguishedName	string	Distinguished name of the zone.
Domain	string	Active Directory domain associated with the zone.
IsBlockGroupInheritance	Boolean	True if groups defined in a parent zone are not inherited, and therefore not visible, in a child zone. This property is only applicable for hierarchical zones.
IsHierarchical	Boolean	True if it is a hierarchical zone.
IsOrphanChildZone	Boolean	True if the zone is a child zone with no parent zone (Hierarchical zone only).
IsSfu	Boolean	True if it is a SFU zone.
Name	string	Name of the zone.
NextGid	long	Next GID value available for assignment to a zone group.

Property	Type	Description
NextUid	long	Next UID value available for assignment to a zone user.
NisDomain	string	NIS domain for SFU zone or agentless mode.
Parent	CdmZone	Parent zone (Hierarchical zone only).
PreferredServer	string	Preferred server to use for committing changes to Active Directory.
ReservedGid	long	Reserved GID values that cannot be assigned to a zone group.
ReservedUid	long	Reserved UID values that cannot be assigned to a zone user.
Schema	string	Schema of the zone.
SfuDomain	string	SFU domain of the zone (SFU zone only).
TenantId	String	The TenantId of the zone
TruncateUserName	Boolean	True if user names longer than 8 characters are automatically truncated for the zone.
Type	string	Type of the zone.
Variables	string[]	Array of runtime variables.

Adding Users in a One-Way Trust Environment

This section explains how to add a user in a one-way trust environment using the authentication and privilege-elevation PowerShell module.

Some operations, such as adding a user to a zone, may require more than one credential. For example, if you want to add a user from one forest to a zone in another forest when there is a one-way trust between the forest, you might need to specify credentials for each forest. This section explains how to add a user in a one-way trust environment when using PowerShell cmdlets.

Using One Account Credential

If you want to add the user targetuser, who has a domain user account in forest2.net to the zone1 in forest1.net, where forest1.net trusts forest2.net (a one-way trust), you must use an account that has the following permissions:

- Permission to add a user to zone1 in forest1.net.
- Permission to read accounts in forest2.net.

If you have a single account with the appropriate permissions–for example, superuser in forest2.net–you can add the targetuser from forest2.net to the zone1 in forest1.net as follows:

```
Set-CdmCredential "forest1.net" "forest2\superuser"
New-CdmUserProfile -Zone "cn=zone1,cn=Zones,dc=forest1,dc=net" -User
"cn=targetuser,cn=Users,dc=forest2,dc=net" -login "UNIXname" -uid nnnnn
```

where UNIXname is the UNIX login name of targetuser and nnnn is the UID of the targetuser.

Using Two Account Credentials

If you do not have a single account with the appropriate permissions in the two forests, adding the targetuser to a zone in another forest will require two account credentials. For example, you must identify accounts with the following permissions:

- An account in forest1.netthat has permission to add a user to zone1 (user1).
- An account in forest2.net that has read permission on forest2.net (user2).

After you identify the accounts with the appropriate permissions–for example, user1 in forest1.net and user2 in forest2.net–you can add the targetuser from forest2.net to the zone1 in forest1.net as follows:

```
Set-CdmCredential "forest1.net" "forest1\user1"
Set-CdmCredential "forest2.net" "forest2\user2"
New-CdmUserProfile -Zone "cn=zone1,cn=Zones,dc=forest1,dc=net" -User
"targetUser@forest2.net" -login "UNIXname" -uid nnnnn
```

where UNIXname is the UNIX login name of targetuser and nnnn is the user's UID.

Using Predefined Scripts to Generate Reports

This section describes the predefined report scripts that are included with the authentication and privilege-elevation PowerShell module and how to configure report output files to generate HTML- and PDF-formatted report files.

Most of the predefined reports in Access Manager Report Center have a corresponding PowerShell script to generate reports from the PowerShell console. When you use a PowerShell script to generate a report, the report content displays as text in the PowerShell console window. You can optionally format the report content as an HTML or PDF file using third-party tools.

Provided Report Scripts

The following report scripts are included with authentication and privilege elevation PowerShell. The scripts are typically installed in the following folder:

`C:\Program Files\Centrify\PowerShell\Centrify.DirectControl.PowerShell\Reports`

For details about script syntax, parameters, and examples, see the script help files. Execute the PowerShell `Get-Help` command to display the help for a script. For example, to display help details for the `ZonesReport.ps1` script, execute the following command from the PowerShell command line:

```
PS> Get-Help .\ZonesReport.ps1 -Detailed
```

This script	Reports this	Equivalent report
AuthorizationReportForComputers.ps1	Lists each computer in the zone and indicates which users are allowed to access each computer. This report applies to classic zones only. This report includes details from the user's UNIX profile for each user listed, including the user's Active Directory user name, UNIX user name, zone, UID, shell, home directory, and primary group.	Classic Zone – Authorization Report for Computers
AuthorizationReportForUsers.ps1	Lists each user account in the zone and indicates which computers each user can access. This report applies to classic zones only. This report includes details from the user's UNIX profile for each user listed, including the user's UNIX user name, zone, UID, shell, home directory, and primary group.	Classic Zone – Authorization Report for Users
ComputerEffectiveAuditLevelReport.ps1	Lists the audit level in effect for all authorized users on computers in each zone. This report applies to hierarchical zones only.	Hierarchical Zone – Computer Effective Audit Level
ComputerEffectiveRightsReport.ps1	Lists the privileges granted on each computer. This report applies to hierarchical zones only.	Hierarchical Zone – Computer Effective Rights
ComputerEffectiveRolesReport.ps1	Lists the roles assigned on each computer. This report applies to hierarchical zones only.	Hierarchical Zone – Computer Effective Roles
ComputerRoleAssignmentsReport.ps1	Lists the computer roles that are defined for each zone. The report includes the users and groups and their associated roles. This report applies to hierarchical zones only.	Hierarchical Zone – Computer Role Assignments

This script	Reports this	Equivalent report
ComputerRoleMembershipReport.ps1	Lists the computer roles that are defined for each computer and the zone to which they belong. This report applies to hierarchical zones only.	Hierarchical Zone – Computer Role Membership Report
ComputersReport.ps1	Lists computer account information for each computer in each zone. The information displayed includes the computer account name in Active Directory, the computer's DNS name, the computer's operating system, and the version of the Delinea Agent for *NIX installed on the computer, if available.	Computers Report
GroupsReport.ps1	Lists group information for each group in each zone. The information that is displayed includes the Active Directory group name, the UNIX group name, the UNIX group identifier (GID), and whether the group is an orphan.	Groups Report
StaleComputersReport.ps1	Lists information about all authentication service-enabled computers that have not changed their password in a specified number of days (90 days by default).	Stale Computers Report
UnixUserEffectiveRightsReport.ps1	Lists the effective rights for each UNIX user on each computer. The report shows the name of the right, its type, and where it is defined. This report applies to hierarchical zones only.	Hierarchical Zone – UNIX User Effective Rights
UserAccountReport.ps1	Lists Active Directory account details for the users that have UNIX profiles in each zone. The report includes the Active Directory display name, logon name, and domain for the account. It also includes the account status, such as the date and time of the account's last logon and whether the account is configured to expire, locked out, or disabled.	User Account Report

This script	Reports this	Equivalent report
UsersReport.ps1	Lists information from the UNIX profile for each user in each zone. The report includes the user's Active Directory user name, UNIX user name, UID, shell, home directory, and primary group.	Users Report
WindowsUserEffectiveRightsReport.ps1	Lists the effective rights for each Windows user on each computer. The report shows the name of the right, its type, and where it is defined. This report applies to hierarchical zones only.	Hierarchical Zone - Windows User Effective Rights
ZoneDelegationReport.ps1	Lists the administrative tasks for each zone and the users or groups (trustees) that have been delegated to perform each task. When you grant administrative rights to designated users and groups, you make them "trustees" with permission to perform specific operations. This report indicates which users or groups have permission to perform specific tasks, such as add groups, join computers to a zone, or change zone properties.	Zone Delegation Report
ZoneRolePrivilegesReport.ps1	Lists the roles that are defined for each hierarchical zone and the rights granted by each of these roles, including where each right is defined.	Hierarchical Zone - Zone Role Privileges Report
ZonesReport.ps1	Lists the zone UNIX properties for each zone. This report includes the zone name, list of available shells, the default shell, the default home directory path, the default primary group, the next available UID, reserved UIDs, the next available GID, and reserved GIDs.	Zones Report

Running Report Scripts

When you perform the steps described in this section, the report content displays as text in the PowerShell console window. To generate formatted reports, see Formatting Reports.

To run a report script:

1. Open the Delinea access module for PowerShell reports.
2. Verify you have permission to execute scripts by running `Get-ExecutionPolicy`. In most cases, the permission to execute scripts is restricted. You can use the `Set-ExecutionPolicy` to allow execution. For example:

   ```
   Set-ExecutionPolicy Unrestricted
   ```

Note: For more information about execution policies and the options available, use the get-help function.

3. Verify that you are in the directory where the report scripts are located. For example:
   `C:\Program Files\Centrify\PowerShell\Centrify.DirectControl.PowerShell\Reports`
4. Execute the report script. For example:
   `.\ZonesReport.ps1`

Formatting Reports

You can use the following cmdlets to format report output so it can be displayed or processed by third-party tools:

- Export-Csv
- Out-GridView
- Format-Table
- ConvertTo-Html

The following sections describe these cmdlets in detail.

Export-Csv cmdlet

Use this cmdlet to format report output as a CSV file. For example, execute the following command to format the output from the `UsersReport.ps1` script as a CSV file:

```
PS> ./UsersReport.ps1 | Export-Csv C:\Report\UsersReport.csv -NoTypeInformation
```

In this example, the output file `C:\Report\UsersReport.csv` is created, and no type information for the input object is provided. After the CSV file is created, you can open it with third-party applications such as Microsoft Excel.

Out-GridView cmdlet

Use this cmdlet to format report output as an interactive table in a grid view window. For example, execute the following command to format the output from the `UsersReport.ps1` script:

```
PS> ./UsersReport.ps1 | Out-GridView
```

Format-Table cmdlet

Use this cmdlet to format report output as a table that is displayed in the PowerShell console window with the selected properties of the object in each column. The object type determines the default layout and properties that are displayed in each column, but you can use the property parameter to select the properties that you want to display. You can specify any of the following parameters on the command line:

- AD User
- Home Directory
- Is Enabled
- Is Orphan
- Primary Group

- Shell
- UID
- UNIX User Name
- Zone

For example, the following command displays the output of `UsersReport.ps1` in a table. The `-GroupBy` option shown here specifies that separate tables are displayed for each zone. Each zone table contains columns for AD User, UNIX User Name, UID, Shell, Home Directory, Is Enabled, Primary Group, and Is Orphan.

```
. PS> ./UsersReport.ps1 | Format-Table "AD User", "UNIX User Name", "UID", "Shell", "Home Directory", "Is Enabled", "Primary Group", "Is Orphan" -GroupBy Zone
```

Depending on your site's zone configuration, this command would result in output similar to the following:

```
PS C:\Progran Files\Centrify\PowerShell\Centrify.DirectControl.PowerShell> ./Use
rsReport | Format-Table "AD User", "UNIX User Name", "UID", "Shell", "Home Direc
tory", "Is Enabled", "Primary Group", "Is Orphan" -groupby Zone


   Zone: ClassicZone-Standard

AD User   UNIX User       UID Shell     Home Dire Is Enable Primary G Is Orphan
          Name                          ctory          d roup
-------   ---------      --- -----     --------- --------- --------- ---------
LAB1\d... deventemp     10001 /bin/bash /home/...      True group999       False
LAB1\I... innert...     10002 /bin/bash /home/...      True                False
LAB1\I... tenpiu...     10006 /bin/bash /home/...      True group999       False
LAB1\I... tenpiu...     10001 /bin/bash /home/...      True                False
LAB1\I... testuser2     10003 /bin/bash /home/...      True                False


   Zone: developers

AD User   UNIX User       UID Shell     Home Dire Is Enable Primary G Is Orphan
          Name                          ctory          d roup
-------   ---------      --- -----     --------- --------- --------- ---------
LAB1\d... deven1        10000 /bin/bash /home/...      True csdevg...      False
LAB1\d... deven2        10002 /bin/bash /home/...      True csdevg...      False
LAB1\d... deventemp     10001 /bin/bash /home/...      True csdevg...      False
LAB1\Jim  jin           10005 /bin/bash /home/jin      True csdevg...      False
LAB1\Joe  joe           10004 /bin/bash /home/joe      True csdevg...      False
          lab1c1...     10009 /bin/bash /home/...      True csdevg...      False
LAB1\s... sally         10003 /bin/bash /home/...      True csdevg...      False
LAB1\I... tenpiu...     10006 /bin/bash /home/...      True csdevg...      False
          tenpiu...     10007 /bin/bash /home/...      True csdevg...       True
          tenpus...     10008 /bin/bash /home/...      True csdevg...       True


   Zone: testers

AD User   UNIX User       UID Shell     Home Dire Is Enable Primary G Is Orphan
          Name                          ctory          d roup
-------   ---------      --- -----     --------- --------- --------- ---------
          tempor...     10001 /bin/bash /home/...      True tempgr...       True
LAB1\t... testen1       10000 /bin/bash /home/...      True tempgr...      False


   Zone: testzone

AD User   UNIX User       UID Shell     Home Dire Is Enable Primary G Is Orphan
          Name                          ctory          d roup
-------   ---------      --- -----     --------- --------- --------- ---------
LAB1\I... tempuser                                 True                False
LAB1\I... testuser1 ...737880 %{shell}  %{home...      True                False

PS C:\Progran Files\Centrify\PowerShell\Centrify.DirectControl.PowerShell>
```

Note: If the results are too wide to display in the PowerShell console default window size, you can change the PowerShell screen size, and enable some arguments (such as wrap or autosize) provided by this cmdlet.

ConvertTo-Html cmdlet

Use this cmdlet to format report output as an HTML file. This cmdlet returns the result to the PowerShell console window. You can then redirect the result to an HTML file by using the cmdlet `Out-File`, so that you can read the output using a Web browser. The HTML file created by this cmdlet uses the style sheet defined in the `report.css` file that is included with authentication and privilege elevation PowerShell.

For example, the following command converts the results of the `UsersReport.ps1` script into HTML using the style defined in report.css, and writes the resulting HTML to the output file `C:\Report\UsersReport.html`.

```
PS> .\UsersReport.ps1 | ConvertTo-Html -CssUri report.css | Out-File
C:\Report\UsersReport.html
```

Generating a PDF report

Overview

This section describes how to use the PDFCreator third-party tool to generate PDF output from a report script. The general steps are as follows:

1. Install the PDFCreator third-party tool.
2. Generate HTML output from a report script using the `ConvertTo-Html` cmdlet.
3. Configure the PDFCreator printer that will convert the HTML output file into a PDF file.
4. Direct the HTML output file to the PDFCreator printer to generate the PDF file.

Procedure Details

The following steps describe how to generate PDF output from the `ZonesReport.ps1` script.

1. Note the following
   - You must have administrator privileges to perform these steps.
   - Unless otherwise noted, you perform the steps described here in the PowerShell console window.
   - In this example, the PDF printer that converts HTML to PDF is named "PDFCreator." If the printer has a different name in your environment, use your printer's name.
2. Install PDFCreator from pdfforge.
3. Generate HTML output from the `ZonesReport.ps1` script by executing the following command in the PowerShell console:

   ```
   .\ZonesReport.ps1 | ConvertTo-Html -Head "<Style>$(Get-Content .\Report.css)</Style>" |
   Out-File c:\Reports\ZonesReport.html
   ```

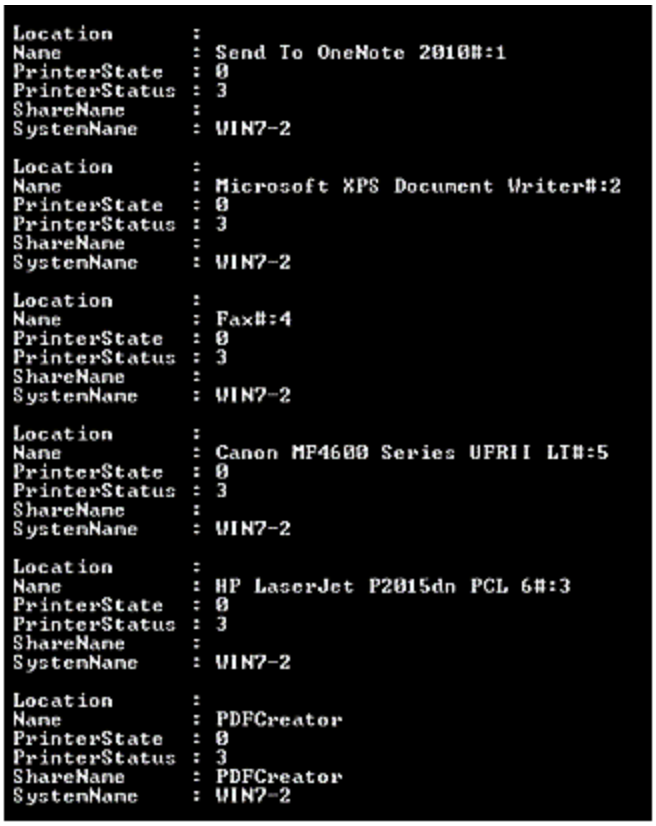
   When you execute this command, the file `c:\Reports\ZonesReport.html` is created using the styles in `Report.css`.
4. Specify PDFCreator as the default printer:
   a. Execute the following command to get all installed printers:

      ```
      $printers = gwmi win32_printer
      ```

   b. Run the following variable to list the printers:

      ```
      $printers
      ```

   c. In the list of printers, note the position of the PDFCreator printer in the list. For example, in the following list of printers, PDFCreator is the sixth printer listed:



d. Make PDFCreator the default printer. In this example, because PDFCreator is the sixth printer on the list, you would execute the following command:

   ```
   $printers[5].SetDefaultPrinter()
   ```

e. Ensure PDFCreator is the default printer by clicking **Devices and Printers** on the Windows Start Menu. If PDFCreator is not the default printer, you can make it the default printer there.

5. Configure the auto-save printer settings as follows:

   a. Change the auto-save directory to `C:\Reports`.

   b. Change the auto-save file name to ZonesReport.

   c. Enable the auto-save feature so that there will be no dialog prompts asking for which file name to save.

6. Perform the following steps to configure the registry to implement these changes. These steps assume that the default registry path is HKCU:\Software\PDFCreator\Program. If your registry path is different, change these commands as appropriate for your environment.

   a. Execute the following command to change the auto-save directory to `C:\Reports`:

      ```
      Set-ItemProperty -Path "HKCU:\Software\PDFCreator\Program"  -Name "AutoSaveDirectory" -Value "C:\Reports"
      ```

   b. Execute the following command to change the auto-save file name to ZonesReport:

      ```
      Set-ItemProperty -Path "HKCU:\Software\PDFCreator\Program" -Name "AutoSaveFileName" -Value "ZonesReport"
      ```

c. Execute the following command to enable the auto-save feature:

```
Set-ItemProperty -Path "HKCU:\Software\PDFCreator\Program" -Name "UseAutoSave" -Value
"1"
```

7. Use Windows Internet Explorer to print the HTML file that you created with the default (PDFCreator) printer. This creates the PDF file.
8. Create and run the following script in the PowerShell console window. The script performs the following tasks:
   a. Creates an IE object and stores it into the $ie variable.
   b. Sets IE output to not display on the screen. This part is optional–if you want IE output to display, you can omit this in the script.
   c. Instructs the $ie object to read the HTML content from the location `C:\Reports\ZonesReport.html` (the HTML file that you created earlier).
   d. Prints the content of $ie using default printer (PDFCreator), resulting in the generation of the PDF file.
9. The recommended script is as follows:

```
$ie = New-Object -com "InternetExplorer.Application"
$ie.Visible = $false`
$ie.Navigate("C:\Reports\ZonesReport.html")
while ( $ie.busy ) { Start-Sleep -second 1 }
  $ie.ExecWB(6,2)
while ( $ie.busy ) { Start-Sleep -second 1 }
  $ie.quit()
```

Note: This script is specific to the example used in this procedure. If you changed any of the steps in this procedure because of differences in your environment, you might have to make corresponding changes in the script shown.