SOP Changing the spool directory

By default, Windows uses the folder %SystemRoot%\System32\spool\PRINTERS to store spooled jobs. On our imaginary print server, PSRV, this folder is   
C:\Windows\System32\spool\PRINTERS. In some cases, particularly when your users   
generate large amounts of printed output, this folder and the volume could become full,   
which is not a good thing. To help you avoid issues, you can move the default spool   
directory to a different folder (for example, C:\Spool), or you could move the spool folder to a folder on another volume (for example, E:\Spool).

Getting ready

There are two ways you can change the spool directory. The first way is to use the classes inside the .NET Framework's System.Printing namespace to update the folder name. The second, and probably the simplest, way is to update the registry with the folder to use for spooling. This recipe shows both methods.

How to do it:

First, let's look at how you change the spool folder using the .NET Framework:

1.Load the System.Printing namespace and classes:

Add-Type -AssemblyName System.Printing

2.Define the required permissions—that is, the ability to administrate the server:

$Permissions = [System.Printing.PrintSystemDesiredAccess]::AdministrateServer

3.Create a PrintServer object with the required permissions:

$Ps = New-Object -TypeName System.Printing.PrintServer -ArgumentList $Permissions

4.Update the default spool folder path:

$Newpath = 'C:\Spool'   
 $Ps.DefaultSpoolDirectory = $Newpath

5.Commit the change:

$Ps.Commit()

6.Restart the Spooler to accept the new folder:

Restart-Service -Name Spooler

7.Once the Spooler has restarted, view the results:

New-Object -TypeName System.Printing.PrintServer | `

Format-Table -Property Name, DefaultSpoolDirectory

Another way to set the Spooler directory is by directly editing the registry as follows:

8.First stop the Spooler service:

Stop-Service -Name Spooler

9.Set the spool directory registry setting:

PS C:\foo> $RPath = 'HKLM:\SYSTEM\CurrentControlSet\Control\Print\Printers'

$Spooldir = 'C:\SpoolViaRegistry'

Set-ItemProperty -Path $RPath -Name DefaultSpoolDirectory -Value 'C:\SpoolViaRegistry'

10.Restart the Spooler:

Start-Service -Name Spooler   
11.View the results:

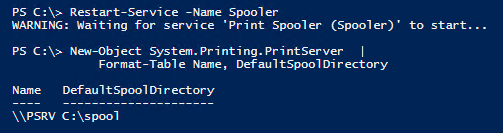
New-Object -TypeName System.Printing.PrintServer | Format-Table -Property Name,DefaultSpoolDirectory

How it works:

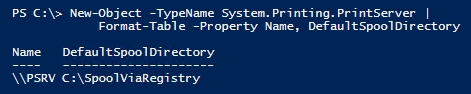
The .NET Framework's System.Printing namespace contains some useful printing-   
related classes and enums, some of which you use in this recipe. PowerShell does not load   
this namespace by default. You load it in *step 1*, using the Add-Type cmdlet, which   
produces no output.

In *step 2*, you create a variable, $Permissions, that holds the print permissions you   
need-namely the ability to administer the print server. In *step 3*, you instantiate a   
PrintServer object with the permission to administer the print server. These permissions are separate from normal administrative privileges. Even running the commands in an   
elevated PowerShell console requires you to create permissions, as you can see here.

In *step 4*, you change the Spool folder to the in-memory PrintServer object, and then in   
*step 5*, you commit the update. In *step 6*, you restart the Spooler, and then, in *step 7*, observe the results from changing the Spooler folder. The output from *step 6* and *step 7* looks like   
this:



The second and simpler method involves just updating the registry value entry that holds the spool folder name (and restarting the Spooler). To do this, in *step 8*, you stop the Spooler, and in *step 9*, you update the registry value that the Spooler system uses for its spool folder. Note that you do *not* have to do *steps 1-7* to use the second method! In *step 10*, you restart the Spooler service, which now uses the new Spool folder. Finally, in *step 11*, you view the results of changing the Spool folder, which looks like this:



Note that the two methods you use in this recipe use different folder names for illustration. The folder name may not be appropriate for your installation. In production, you should   
also consider moving the Spool folder to a separate volume to avoid running out of space   
on the system volume.

This recipe makes use of the underlying .NET System.Printing namespace instead of just commands from the PrintManagement modules. This approach has value in many other   
places inside Windows. In general, the advice is to use cmdlets where/when you can and   
only then dip down into either the .NET Framework or the CIM/WMI namespaces and   
classes.