

Christmas Cheer Laser, part 3

PowerShell Drives

PowerShell has broadened the concept of drives to include containers in general (not to be confused with Docker containers.) PowerShell treats many things that have the tree structure you are used to seeing in directories the same way as it treats directories. This is why PowerShell calls the `dir` or `ls` cmdlet “`Get-ChildItem`”. You can get the listing of Environment variables the same way you list a directory.

```
Get-ChildItem Env:
```

On a Windows machine, the hard drives (C:, D:, etc.), the Environment variables (Env:), the registry (HKCU: and HKLM:), the certificate store (Cert:) are all accessible with the `Get-ChildItem` cmdlet. Many features of other Microsoft products like Active Directory and SQL server are accessible this way as well.

The aliases for `Get-ChildItem` on a Windows machine are `dir`, `gci`, and `ls`, but `ls` is missing on this terminal.

Answers to Previous Questions

- 3) Find the message (and a parameter, too) in the command History

`Get-History` shows us most of what we need, but line 9 is truncated.

```
PS /home/elf> Get-History

Id CommandLine
--
1 Get-Help -Name Get-Process
2 Get-Help -Name Get-*
3 Set-ExecutionPolicy Unrestricted
4 Get-Service | ConvertTo-HTML -Property Name, Status > C:\services.htm
5 Get-Service | Export-CSV c:\service.csv
6 Get-Service | Select-Object Name, Status | Export-CSV c:\service.csv
7 (Invoke-WebRequest http://127.0.0.1:1225/api/angle?val=65.5).RawContent
8 Get-EventLog -Log "Application"
9 I have many name=value variables that I share to applications system wide. At a command...
10 dir
11 gc /home/callingcard.txt
```

Notice that line 7 has the value for angle, `angle?val=65.5`

`Get-History | fl` shows us the entire line.

```
PS /home/elf> Get-History | fl

<snip>

Id          : 9
CommandLine : I have many name=value variables that I share to applications
              system wide. At a command I will reveal my secrets once you Get my
              Child Items.
ExecutingStatus : Completed
```

- 4) The answer to the previous question includes, “name=value variables that I share to applications system wide.” That’s a strong hint to tell you where to look next. The terms “variables”, “share to applications system wide”, and “name=value” are all useful hints.

The name=value variables are environment variables. PATH is a common environment variable. You can see them with
Get-ChildItem Env:
(dir env: or gci env: would also work.)

```
PS /home/elf> Get-ChildItem Env:

Name                           Value
----                           -
_                               /bin/su
DOTNET_SYSTEM_GLOBALIZATION_I... false
HOME                           /home/elf
HOSTNAME                       af9b0c87535c
LANG                           en_US.UTF-8
LC_ALL                         en_US.UTF-8
LOGNAME                         elf
MAIL                           /var/mail/elf
PATH                           /opt/microsoft/powershell/6:/usr/local/sbin:/usr/local/bi...
PSModuleAnalysisCachePath      /var/cache/microsoft/powershell/PSModuleAnalysisCache/Mod...
PSModulePath                   /home/elf/.local/share/powershell/Modules:/usr/local/shar...
PWD                             /home/elf
RESOURCE_ID                    05a5814c-e84f-49cd-b8d6-14605ff520e8
riddle                         Squeezed and compressed I am hidden away. Expand me from ...
SHELL                          /home/elf/elf
SHLV                            1
TERM                           xterm
USER                           elf
USERDOMAIN                     laserterminal
userdomain                     laserterminal
username                       elf
USERNAME                       elf

PS /home/elf>
```

Another ellipsis! Since we know the answer we need is in riddle, we can treat it just as if it were a file:

```
dir env:riddle | fl
```

```
PS /home/elf> dir env:riddle | fl

Name : riddle
Value : Squeezed and compressed I am hidden away. Expand me from my prison and I will
       show you the way. Recurse through all /etc and Sort on my LastWriteTime to
       reveal im the newest of all.

PS /home/elf>
```

Get-Member

PowerShell deals with objects, not just text, passed down the pipeline with the pipe symbol (|). We must have a way to learn what the contents of an object are, and the Get-Member cmdlet does that for us. In this case the riddle tells us we need to sort something on LastWriteTime. Since we are using Get-ChildItem (or dir) we can guess that LastWriteTime is a property of those objects. We can test that by piping the output of Get-ChildItem into Get-Member.

Get-ChildItem | Get-Member

```
PS /home/elf> Get-ChildItem | Get-Member
```

TypeName: System.IO.DirectoryInfo		
Name	MemberType	Definition
----	-----	-----
LinkType	CodeProperty	System.String LinkType{get=GetLinkType;}
Mode	CodeProperty	System.String Mode{get=Mode;}
Target	CodeProperty	System.Collections.Generic.IEnumerable`1[[System.String, System.Private.CoreLib, Version=6.0.0.0, Culture=neutral, PublicKeyToken=7cec75434710809e]]
Create	Method	void Create()
CreateSubdirectory	Method	System.IO.DirectoryInfo CreateSubdirectory(string path)
Delete	Method	void Delete(), void Delete(bool recursive)
EnumerateDirectories	Method	System.Collections.Generic.IEnumerable[System.IO.DirectoryInfo] EnumerateDirectories()
EnumerateFiles	Method	System.Collections.Generic.IEnumerable[System.IO.FileInfo] EnumerateFiles()
EnumerateFileSystemInfos	Method	System.Collections.Generic.IEnumerable[System.IO.FileSystemInfo] EnumerateFileSystemInfos()
Equals	Method	bool Equals(System.Object obj)
GetDirectories	Method	System.IO.DirectoryInfo[] GetDirectories(), System.IO.DirectoryInfo[] GetDirectories(string path)
GetFiles	Method	System.IO.FileInfo[] GetFiles(), System.IO.FileInfo[] GetFiles(string path)
GetFileSystemInfos	Method	System.IO.FileSystemInfo[] GetFileSystemInfos(), System.IO.FileSystemInfo[] GetFileSystemInfos(string path)
GetHashCode	Method	int GetHashCode()
GetLifetimeService	Method	System.Object GetLifetimeService()
GetObjectData	Method	void GetObjectData(System.Runtime.Serialization.SerializationInfo info, System.Runtime.Serialization.StreamingContext context)
GetType	Method	type GetType()
InitializeLifetimeService	Method	System.Object InitializeLifetimeService()
MoveTo	Method	void MoveTo(string destDirName)
Refresh	Method	void Refresh()
ToString	Method	string ToString()
PSChildName	NoteProperty	string PSChildName=depths
PSDrive	NoteProperty	PSDriveInfo PSDrive=
PSIsContainer	NoteProperty	bool PSIsContainer=True
PSParentPath	NoteProperty	string PSParentPath=Microsoft.PowerShell.Core\FileSystem::PSDrive
PSPath	NoteProperty	string PSPath=Microsoft.PowerShell.Core\FileSystem::PSDrive
PSProvider	NoteProperty	ProviderInfo PSProvider=Microsoft.PowerShell.Core\FileSystem
Attributes	Property	System.IO.FileAttributes Attributes {get;set;}
CreationTime	Property	datetime CreationTime {get;set;}
CreationTimeUtc	Property	datetime CreationTimeUtc {get;set;}
Exists	Property	bool Exists {get;}
Extension	Property	string Extension {get;}
FullName	Property	string FullName {get;}
LastAccessTime	Property	datetime LastAccessTime {get;set;}
LastAccessTimeUtc	Property	datetime LastAccessTimeUtc {get;set;}
LastWriteTime	Property	datetime LastWriteTime {get;set;}
LastWriteTimeUtc	Property	datetime LastWriteTimeUtc {get;set;}
Name	Property	string Name {get;}
Parent	Property	System.IO.DirectoryInfo Parent {get;}
Root	Property	System.IO.DirectoryInfo Root {get;}
BaseName	ScriptProperty	System.Object BaseName {get=\$this.Name;}

There is a property of the object called LastWriteTime.

To answer the riddle, you need to get a listing (recursively, meaning you have to include all sub directories) of the /etc directory, then pipe these results into Sort (Sort-Object) and sort the results using the LastWriteTime property to find the newest file. By default, Sort-Object will put the oldest object at the top and the newest at the bottom. Once you have found the file, you need to expand or decompress the file. PowerShell has a cmdlet for that, you just need to find it.

Question

5) What riddle do you find inside the archive?