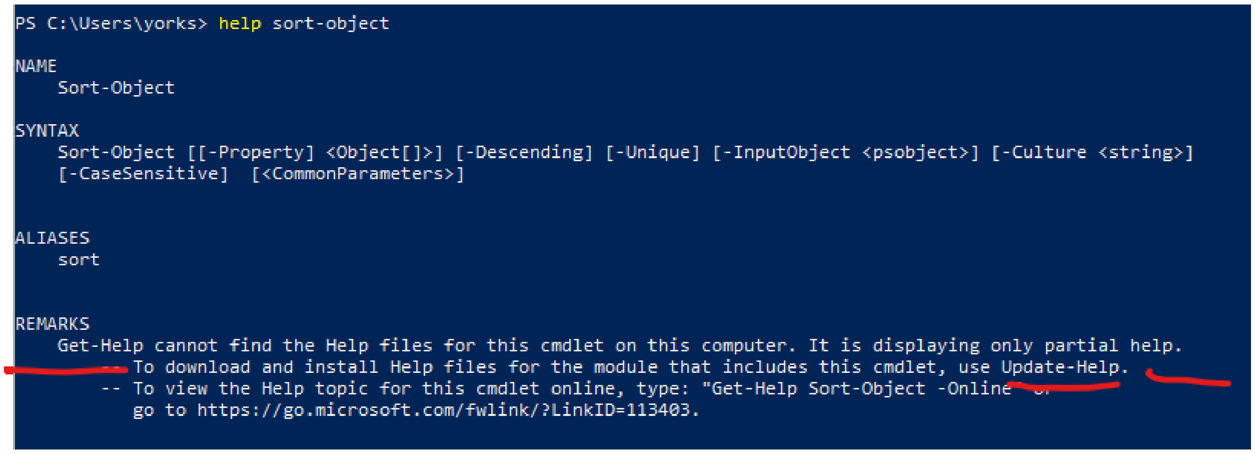
Christmas Cheer Laser, part 4

Get-Help (alias help)

Since we are going to use the `Sort-Object` cmdlet, we might as well learn a little about it. Search engines are most people's primary tools for learning about commands, but PowerShell also includes a help cmdlet called `Get-Help`. The full help is not installed by default on Windows machine. To add it, use `Update-Help` from an elevated terminal (Administrator.)



```
PS C:\Users\yorks> help sort-object

NAME
    Sort-Object

SYNTAX
    Sort-Object [[-Property] <Object[]>] [-Descending] [-Unique] [-InputObject <psobject>] [-Culture <string>]
    [-CaseSensitive]  [<CommonParameters>]


ALIASES
    sort


REMARKS
    Get-Help cannot find the Help files for this cmdlet on this computer. It is displaying only partial help.
        -- To download and install Help files for the module that includes this cmdlet, use Update-Help.
        -- To view the Help topic for this cmdlet online, type: "Get-Help Sort-Object -Online" or
           go to https://go.microsoft.com/fwlink/?LinkID=113403.
```

Unfortunately, this only works on your own host where you have administrator access. We are using someone else's terminal, so we won't be able to install the full help. (I just thought you should know about it.)

Sort-Object (alias sort)

In the help screen above, we see that `Sort-Object` wants a parameter `[[-Property] <Object[]>]` to sort on. The square brackets denote something that is optional, so we could have no parameters and let the cmdlet sort on whatever it wanted to (usually name.) That is silly, so we would use
Sort-`Object -Property LastWriteTime`
for example. Note that `–Property` is also inside square brackets. This means that `–Property` is a positional parameter. It is used so often that the cmdlet will assume you mean `–Property` even if it isn't present. You can save typing by using
`Sort-Object LastWriteTime` or even `sort LastWriteTime`.

If you want to reverse the sort order, you use `–Descending`.

Note: When you see `something[]` in the help file, that means that it can accept an array of `somethings`.

Select-Object (alias select)

We will talk more about `Select-Object` later, but it has useful parameters that will help us now. When you have a lot of output and you only want to see the beginning or end of the data so your screen doesn't fill up, pipe your output into `Select-Object -First` (or `-Last`.) You can control how many lines you see by giving it a number; `Select-Object -First 10` will show the first 10 lines. This is like the Linux `head` command. The Linux `tail` is like `–Last`.

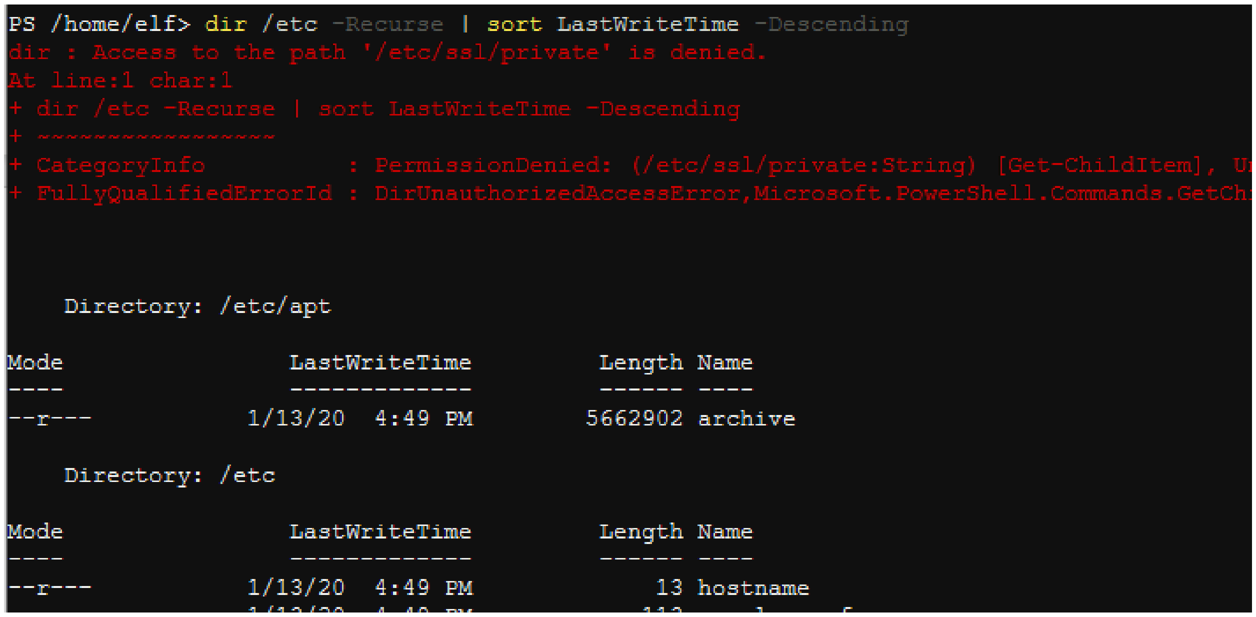
Answer to the Previous Question

5) What riddle do you find inside the archive?

We can do a variation of the command above, but it has problems.

`dir /etc -Recurse | sort LastWriteTime -Descending`

There is a lot of data that scrolls off the top of the screen, and there is nasty red error text. The reason for the error is that the `Get-ChildItem` (`dir`) cmdlet tried to look in places where it didn't have permission.

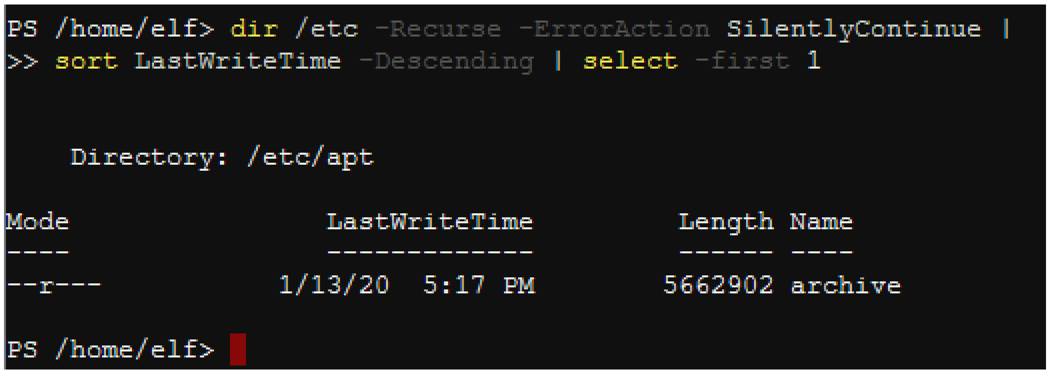


```
PS /home/elf> dir /etc -Recurse | sort LastWriteTime -Descending
dir : Access to the path '/etc/ssl/private' is denied.
At line:1 char:1
+ dir /etc -Recurse | sort LastWriteTime -Descending
+ ~~~~~~~~~~~~~~~~~
+ CategoryInfo          : PermissionDenied: (/etc/ssl/private:String) [Get-ChildItem], U
+ FullyQualifiedErrorId : DirUnauthorizedAccessError,Microsoft.PowerShell.Commands.GetCh



    Directory: /etc/apt

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
--r---          1/13/20   4:49 PM        5662902 archive

    Directory: /etc

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
--r---          1/13/20   4:49 PM             13 hostname
```

We can remove the error text by telling PowerShell to ignore it. We add this:

`-ErrorAction SilentlyContinue`

To just see the first line of the result, we can pipe into `Select-Object` (alias `select`) with the `-First 1` option. Since the command is long, I hit Enter after the pipe, and PowerShell allowed me to continue.



```
PS /home/elf> dir /etc -Recurse -ErrorAction SilentlyContinue |
>> sort LastWriteTime -Descending | select -first 1


    Directory: /etc/apt

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
--r---          1/13/20   5:17 PM        5662902 archive

PS /home/elf>
```

Success!

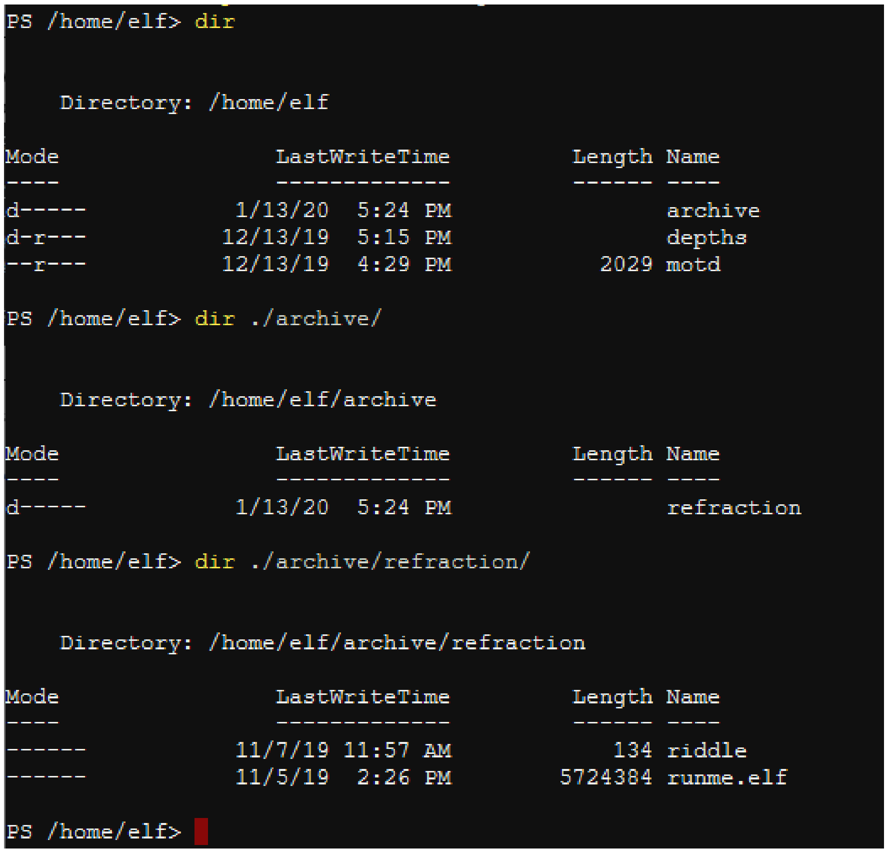
To expand the archive we use, surprise, `Expand-Archive`. In its long form the command would be

`Expand-Archive -Path /etc/apt/archive -DestinationPath ./archive`

or in short form

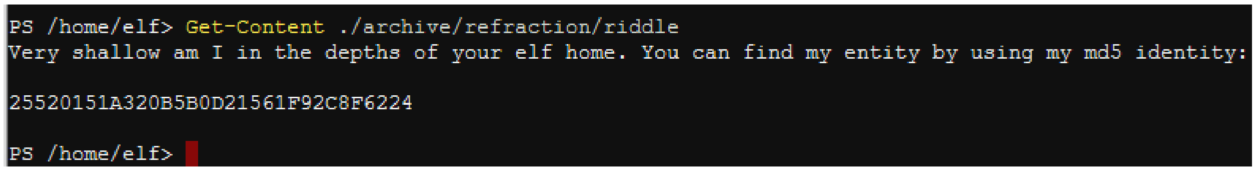
`Expand-Archive /etc/apt/archive ./archive`

Let's see what we expanded.



```
PS /home/elf> dir


    Directory: /home/elf

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----          1/13/20   5:24 PM                archive
d-r---         12/13/19   5:15 PM                depths
--r---         12/13/19   4:29 PM           2029 motd

PS /home/elf> dir ./archive/


    Directory: /home/elf/archive

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----          1/13/20   5:24 PM                refraction

PS /home/elf> dir ./archive/refraction/


    Directory: /home/elf/archive/refraction

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
------          11/7/19 11:57 AM            134 riddle
------          11/5/19   2:26 PM        5724384 runme.elf

PS /home/elf>
```

They are making us work, for sure. Remember the runme.elf file—we'll need to come back to that. The .elf extension normally indicates a Linux executable file.



```
PS /home/elf> Get-Content ./archive/refraction/riddle
Very shallow am I in the depths of your elf home. You can find my entity by using my md5 identity:

25520151A320B5B0D21561F92C8F6224

PS /home/elf>
```

Another option in `Get-ChildItem` (`dir`) to go with `-Recurse` is `-Depth`, and it specifies how many subdirectory levels to look at. You can use `-Depth` without `-Recurse`.

PowerShell has a cmdlet to take hashes of files, and it can use several different algorithms.

Where-Object (aliases where and ?)

Using the `Where-Object` cmdlet is like putting an if statement in the middle of the pipeline. Objects which meet the requirements of your test continue down the pipeline; those which do not are dropped. The general syntax is
`some objects | Where-Object {someTest -eq True} | true, keep on going.`

Note that the test is enclosed in curly braces, which denote a script block. Also note that the comparison operators are things like `-eq, -ne, -gt,` and the like. See this site for a list of operators: https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_comparison_operators?view=powershell-7

Another useful thing when using the pipeline is the `$_` automatic variable. I think it was taken from Perl, and the concept is simple; it refers to whatever object was passed down the pipeline. You can reference properties of that object with `$_.property`. For example, if you are only interested in services that are stopped, you could do this.

```
Get-Service | Where-Object {$_.Status -eq "Stopped"}
```

`Get-Service` puts an object into the pipeline for every service. The `Where-Object` cmdlet only lets those that have a `Status` of `Stopped` continue.

Question

6) Do a search of `/home/elf` to a "shallow" depth. Compute the MD5 hash of all the files you find, and output the name and path of the file that has a hash of `25520151A320B5B0D21561F92C8F6224`