Chapter 26 Lab

Listing 26.2 shows a complete script. See if you can figure out what it does, and how to use it. Can you predict any errors that this might cause? What might you need to do in order to use this in your environment? Note that this script should run as-is… but, if it doesn’t on your system, do you think you can track down the cause of the problem? Keep in mind that you’ve seen most of these commands – and for the ones you haven’t there are the PowerShell help files. Those files’ examples include every technique shown in this script.

The script file seems to define 2 functions which won’t do anything until called. At the end of the script is a command, Get-LastOn, which is the same name as one of the functions so we can assume that is what is executed. Looking at that function it has a number of parameter defaults which explains why nothing else needs to be called. The comment based help also explains what the function does. The first part of this function is using Get-Eventlog.

$eventsAndIDs = Get-EventLog -LogName security -Newest $Newest | Where-Object {$\_.instanceid -eq $logonEventNum -or $\_.instanceid -eq $logoffEventNum} |

Select-Object -Last $maxIDs -Property TimeGenerated,Message,ComputerName

If this was a new cmdlet, we would look at help and examples. The expression seems to be getting the newest security eventlogs. $Newest comes from a parameter and has a default value of 5000. These eventlogs are then filtered by Where-Object looking for two different event log values, also from the parameter.

Next it looks like something is done with each event log in the foreach loop. Here’s a potential pitfall: if the eventlog doesn’t have any matching errors the code in this loop will likely fail unless it has some good error handling.

In the foreach loop it looks like some other variables are getting set. The first one is taking the event object and piping it to something called parseEventmessage. This doesn’t look like a cmdlet name but we did see it as one of the functions. Jumping to it, we can see that it takes a message as a parameter and splits each one into an array. We might need to research the –Split operator.

Each line in the array is processed by another ForEach loop. It looks like lines are split again and there is a Try/Catch block to handle errors. Again, we might need to read-up on that to see how it works. Finally there is an IF statement where it appears that if the split up strings are not empty, then a variable called $props is created as a hash table or associative array. This function would be much easier to decipher if the author had included some comments. Anyway, the parsing function ends by calling New-Object, another cmdlet to read up on.

This function’s output is then passed to the calling function. It looks like the same process is repeated to get $domain.

Oh, look another hash table and New-Object but by now we should understand what the function is doing. This is the final output from the function and hence the script.