Updated Version of the Graph User Statistics Script Available

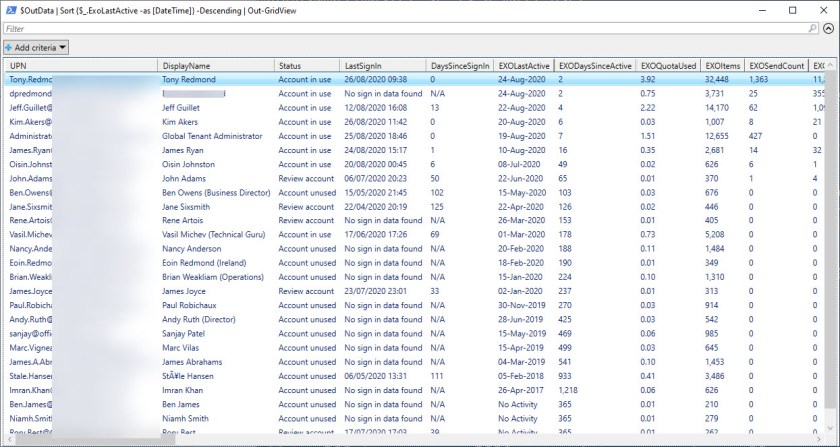
Posted on [**August 29, 2020**](https://office365itpros.com/2020/08/29/updated-version-graph-user-statistics-script-available/) by [**Tony Redmond**](https://office365itpros.com/author/thoughtsofanidlemind/)

The GetGraphUserStatisticsReport.PS1 Script

Last June, I wrote about a [**PowerShell script to interrogate the Microsoft Graph to retrieve usage data**](https://petri.com/graph-powershell-office365-usage) from workloads like Exchange Online, SharePoint Online, and Teams. Some of this data is available via PowerShell cmdlets like *Get-ExoMailboxStatistics* and *Get-SPOSite*, but using the Graph is usually faster.

Find Inactive Accounts

The idea behind the script is that if you retrieve usage information from a bunch of workloads, you have the basis to figure out what accounts are in active use. Potentially, you might even find some accounts that haven’t been used for a while that you can archive or remove to save some Office 365 licenses. Figure 1 shows the script output as viewed through the Out-GridView cmdlet.

Figure 1: Output from GetGraphUserStatisticsReport.PS1

The original script worked, but it had some performance issues once people tried to run it in tenants with thousands of accounts. Long-running PowerShell scripts are not usual, but the issues people ran into pointed to unreasonable delays.

GitHub Contributors

One of the advantages of storing scripts on GitHub is that others can work with the code to detect where problems might lurk. In this case, suspicion fell on how the script processed data in the array storing information fetched from the Graph. It was suggested that better results might be achieved by changing to a [**PowerShell DataTable object**](https://petri.com/dancing-on-the-table-with-powershell). I did some work to test the assertion but didn’t see a huge gain, probably because I don’t have a big enough tenant to test.

In any case, a simpler fix became obvious. Instead of scanning the big list containing all the records for the records (usually six) for each user, it’s much better to extract the records for the user into an array and use that array. The change is made in version 1.2 of the script, which is now [**available to download from GitHub**](https://github.com/12Knocksinna/Office365itpros/blob/master/GetGraphUserStatisticsReport.PS1). (update: The current version of the script is 2.0).

It would be helpful if those who have admin permissions in tenant with more than a few thousand accounts could test the script (after carefully checking that the script is OK to run in your tenant, creating the necessary registered app in Azure AD, and so on). If you find problems, record them in GitHub and the community will have a chance to work the issues.

V1.3 Update

We’ve just updated the script to V1.3 to incorporate another speed tweak. The [**Where method**](https://mcpmag.com/articles/2015/12/02/where-method-in-powershell.aspx) is more appropriate and faster when you only want to extract data from a set and don’t need to feed that data through the pipeline. The script extracts records for each user to create an assessment of the user’s activity and using the Where method is faster… A good lesson learned!