# Invoke-M365PSTImport

## Description of script & scope

This script was written to address a specific need to import PST files in an automated fashion. Unfortunately, the free Azure storage space provided by Microsoft does not allow you to script the PST import process completely, so a different solution was needed

This script will create storage space in azure and upload the files to this blob. **This does mean that costs will be incurred for the storage, and manual deletion of the PST files is needed after verification that they were imported successfully.**

## Required prerequisites

* User account which is member of Mailbox Import Export role in Exchange Online
* Right to create Azure resource group and storage account
* Run script in local admin context to allow for the automated installation of required PowerShell modules.

## Usage

### Creation of Azure blob

.\invoke-M365Import.ps1 -ConfigureBlob -StorageAccountName <Contoso> -StorageAccountContainer <PSTUploads> -TokenExpireDate “02/01/2019” -StorageLocation ‘West Europe’ -Runlog C:\Logs\M365Import.log

|  |  |  |
| --- | --- | --- |
| Parameter | Required | Description |
| ConfigureBlob | No | Indicates that we want to configure the storage Blob in Azure |
| StorageAccountName | Yes | Forms part of the FQDN: Contoso.blob.core.windows.net |
| StorageAccountContainer | Yes | Forms part of the URI:  https://Contoso.blob.core.windows.net/PSTUpload |
| TokenExpireDate | Yes | Date the SAS token will expire |
| StorageLocation | Yes | Region the storage blob will be located. Can be any of the following:   * "West Europe" * "North Europe" * "East US 2" * "Central US" * "South Central US" * "West US" * "East US" * "Southeast Asia" * "East Asia" * "Japan West" |
| Runlog | No | Path to the log file. If not specified, it will be created in the script root directory |

When running the ConfigureBlob set, output will be returned in the powershell window. You will have to save this output manually at this time. Following information will be returned:

* Storage Account URI
* SAS Token: To be used when importing PST files
* Storage Key: To be used when uploading PST files
* SAS URI: Not used at this time

### Upload of PST files

.\invoke-M365Import.ps1 -UploadPST -AzureBlobStorageAccountUri <Storage Account URI> -AzureSharedAccessSignatureToken <Storage Key> -AZCopyVerbose -Recurse -Path C:\Test\ -CSVPath C:\Test\ -EmailAddress John.Doe@contoso.com

|  |  |  |
| --- | --- | --- |
| Parameter | Required | Description |
| UploadPST | No | Indicates we want to upload PST files |
| AzureBlobStorageAccountURI | Yes | URI to where the PST files will be uploaded |
| AzureSharedAccessSignatureToken | Yes | Storage key to access Azure storage blob |
| Recurse | No | Look for files recursively |
| Pattern | No | Look for a given pattern |
| AZCopyVerbose | No | Will output logging to the screen |
| Path | Yes | *Alias: PSTPath*  Path to where the PST files are located |
| CSVPath | Yes | Location where the CSV mapping file will be stored |
| EmailAddress | Yes | *Alias: Email*  *Alias: mail*  *Alias: UPN*  Email address of the user we are currently processing PST files for |
| Runlog | No | Path to the log file. If not specified, it will be created in the script root directory |

Output:

Mapping CSV file

### Importing of PST files

.\invoke-M365Import.ps1 -ImportPST -AzureBlobStorageAccountUri <Storage Account URI> -AzureSharedAccessSignatureToken -CSVPath C:\Test\ -Batchname “Importing!”

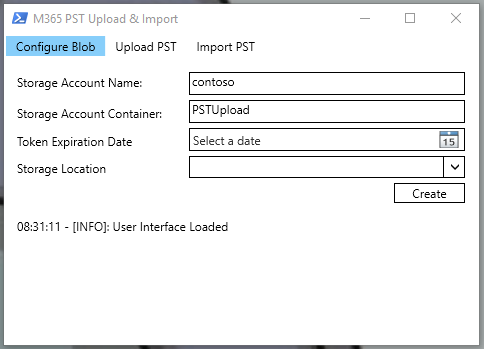
|  |  |  |
| --- | --- | --- |
| Parameter | Required | Description |
| ImportPST | No | Indicates we want to import PST files |
| AzureBlobStorageAccountURI | Yes | URI to where the PST files will be uploaded |
| AzureSharedAccessSignatureToken | Yes | SAS Token to access Azure storage blob. |
| CSVPath | Yes | Location where the CSV Mapping file is stored |
| BatchName | Yes | Name of the batch |

## Graphical User Interface

The GUI version of the script can be launched by running the “Invoke-GUI.ps1” script in the root of the script folder.

### Configure Blob tab

In this window you can create the azure blob.



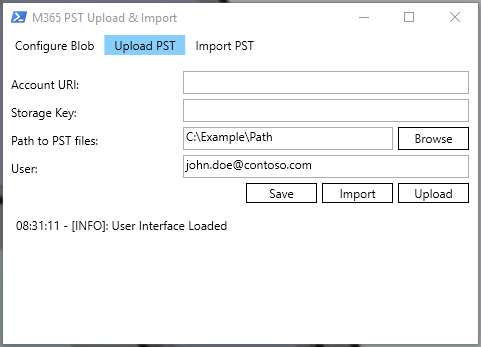
|  |  |  |
| --- | --- | --- |
| Parameter | Required | Description |
| Storage Account Name | Yes | Forms part of the FQDN: **Contoso**.blob.core.windows.net |
| Storage AccountContainer | Yes | Forms part of the URI:  https://Contoso.blob.core.windows.net/**PSTUpload** |
| Token Expire Date | Yes | Date the SAS token will expire |
| StorageLocation | Yes | Region the storage blob will be located |

#### Create button

Starts the process to create the Azure blob. This will also automatically populate most fields in the GUI.

### Upload PST

The upload PST Tab is used for uploading PST files to the previously created Azure blob.



|  |  |  |
| --- | --- | --- |
| Parameter | Required | Description |
| Account URI | Yes | URI to where the PST files will be uploaded |
| Storage Key | Yes | Storage key to access Azure storage blob |
| Path to PST files | Yes | Path to where the PST files are located |
| User | Yes | Email address of the user we’re currently uploading PST files for. |

#### Browse button

Opens a file browser dialog to select the path of the PST files.

#### Save button

Saves all settings to an XML file in the root directory of the script. Due to the sensitive nature of the content (Storage key and SAS token), it is your responsibility to secure access to this file.

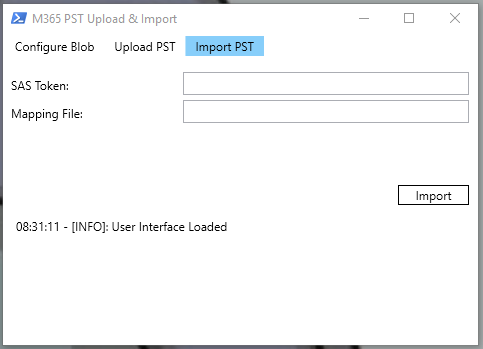
#### Import button

Imports the settings from the saved XML file

#### Upload button

Starts the PST upload process.

### Import PST



|  |  |  |
| --- | --- | --- |
| Parameter | Required | Description |
| SAS Token | Yes | SAS Token to access Azure storage blob. |
| Mapping File | Yes | Path to CSV file containing the mapping of file name and user.  *This file is generated* ***automatically*** *in the root script folder (.\Mapping.csv)* |

#### Import Button

Starts the process to import the PST files previously uploaded. A new import request will be created for each PST file in the mapping file.

## Known limitations

Not yet documented

## Changelog

December 18th, 2018

* Release of first version