Profile Manager Deployment & Use

What is Profile Manager?

Profile Manager is an application that is designed to update existing end user Outlook profiles of migrated mailboxes. Without such an application, you would be forced to visit user desktops to manually create new Outlook profiles for each mailbox that was migrated. In an organization of several hundred or thousands of mailboxes, this can be a daunting task, if not insurmountable. The Profile Manager simplifies this task by allowing you to automate the updates and only for accounts that have migrated.

End User Desktop System Requirements

Profile Manager is a self-contained application that runs on user desktops that run a Windows operating system that has a version of Microsoft Outlook installed. The list below is the minimum systems requirements for Profile Manager to execute. It does not require pre-installation and is not required to be copied locally to the user's computer. The base requirement is that the application is in the user's context, meaning not elevated or as an admin.

Operating system: Windows operating systems – Minimum supported version is Windows XP
Microsoft .Net Framework: Users' desktops must have at least one or more of the following versions of the .NET Framework: 2.0, 3.0, 3.5, 3.5.1, 4.0, 4.5 or later.

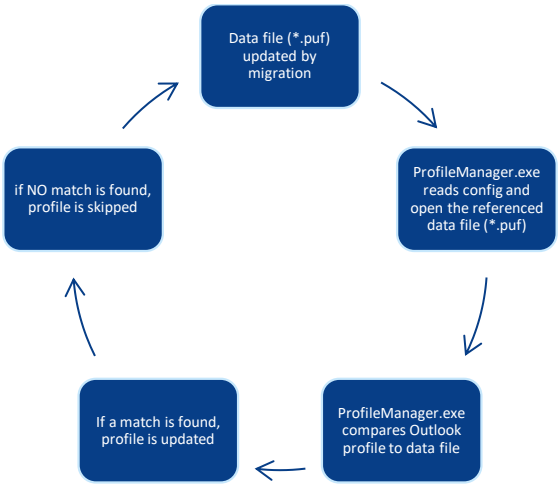
System Requirements Details: Early rollout and use of the Priasoft Profile Manager is essential to success. When Profile Manager runs before migration, it will be able to report as to the requirements necessary of a given user's computer. This helps identify hosts that do not have a supported version of the .NET Framework, do not have outlook installed, or have not Outlook profiles configured. Early deployment and automation of the Profile Manager provides the following additional benefits:

- Proper function. The app is able to run and has the appropriate pre-requisites such as the .NET framework.
- Has Outlook installed.
- Has one or more Outlook profiles configured.
- Provides early feedback to know that all users expected to run the application are, in fact, running the tool.
- Allows for early detection of runtime issues before any migration would occur.

When pre-requisites are not found on a user's desktop, such is logged to the configured log location (by default in the same location as the ProfileManager.exe application) so that such can be identified and addressed before migration.

How it works

The Profile Manager application works in combination with 2 files: a configuration file (prof.config) and a data file (mbm.puf). The configuration file contains location information for the data file and the log output directory, and determines the behavior of Profile Manager when it runs. When the application executes, it looks in the current working directory (see: Working Directory) for a file named **prof.config**. The configuration file is a text file in an INI (see: INI File) format. The Profile Manager parses the configuration file and collects the value for the location of the data file (typically named mbm.puf; PUF equates to Profile Update File). PROFILE MANAGER then scans each Outlook profile under the user's current login and compares details of the profile(s) to details in the data file - if a matching profile is found, the profile is updated based on the settings in the configuration file.



Technical Details

Profile Manager is a standalone application that has only a dependency on Outlook and Microsoft .NET (no Java, or other frameworks are required). Profile Manager uses the system registry to analyze and update Outlook profile(s). It is important to note that only one application can have read+write access to an Outlook profile. As such, it is important to run PROFILE MANAGER when no other MAPI applications are running (such as Outlook). Furthermore, it equally important to make sure end users are not running Outlook during the migration to avoid potential issues with newer versions of Outlook that heavily rely on Auto Discover. If Outlook clients are left running during the migration, it's possible the Outlook client will negativity react to the changes in the environment before Profile Manager has a chance to 'fix-up' the profile before its next use.

An Outlook profile is stored in the registry – HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows Messaging Subsystem\Profiles or for Outlook 2016 based clients in HKEY_CURRENT_USER\Software\Microsoft\Office\16.0\Outlook\Profiles. Note that the profile is stored in the **Current User** section of the registry. As such, Profile Manager must run as the current user in order to update the profile. This means that Profile Manager must be executed as the user, whether directly by the end user or by automation. Furthermore, by design, Profile Manager only updates existing profiles and will not create new profiles.

Supported versions of Outlook

Profile Manager currently supports Outlook 98 and above, including Outlook 2016 and also 32 and 64 bit versions of Outlook. Note that only certain versions of Outlook are supported by Microsoft for use on specific versions of Exchange (e.g. Outlook 2000 will not connect to Exchange 2010). Please make sure to identify this prior to going into production and make the necessary changes to support the target version of Exchange.

Process Details

When PROFILE MANAGER is run, without command-line parameters, it looks for **prof.config** in the same location as Profile Manager Application, and if found, parses the contents of the file. One of the values in the config file it needs is the location of the data file:

```
puffile=\\servername\sharename\foldername\mbm.puf
```

The above line indicates the location for the data file. The data file is created and updated by Mailbox Migration Manager. (Note: If the data file does not exist, the first migration process will create it and will subsequently append to the file afterwards. The Mailbox Migration Wizard does not create this file.) When a mailbox **successfully** completes (failed mailboxes do not write to the data file), it writes one line of data to the data file. The line added contains X500 addresses (see: msdn.microsoft.com) of the source and target mailboxes as well as server names and the primary SMTP address and display name of the user's mailbox. As Profile Manager enumerates each Outlook profile it gathers the X500 address of the primary and any secondary (see: office.microsoft.com) mailboxes. PROFILE MANAGER searches the data file for the X500 address (which is in the first field of each line in the data file), and

with the first match found, attempts to update the profile with the new information about the location of the migrated mailbox. Upon successful update, the data in the profile is changed and therefore if PROFILE MANAGER is run again it will not update again because the value in the profile will not be found in the data file.

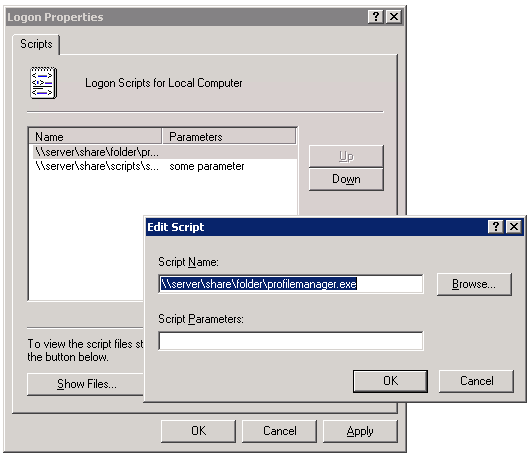
Deployment

Deployment of the Profile Manager is left to the implementer. Priasoft provides the application to update Outlook profiles, but does not provide the automation mechanism. This allows for the broadest amount of flexibility with regards to handling custom business situations.

Automation is preferred where possible. The most common way to automate the update of Outlook profiles is via a logon script. In most organizations, users logon to a Windows Domain within which it is easy to implement such a script. When leveraging a logon script to automate profile updates, it is **highly recommended** to have Profile Manager run as the very first item in the script. The reason is to avoid any chance of some process in the logon script interfering with the update process. Remember from information given above that only one application can make changes to an Outlook profile at a time; as such if some other application is started in the logon script process that uses an Outlook profile, such can prevent PROFILE MANAGER from updating the same profile. Automation can also be done via any other desktop automation system your organization may have. Some examples are Altiris and Microsoft's SCCM (or the older SMS). The key requirement is that the application is run in the user's desktop session (e.g. running as the user).

Note, the simplest way to implement a logon script for profile updates is to use a Domain Policy and direct the policy to run the PM application directly – there is no requirement to call the PM from a batch file or other script language.

Notice below that profileManager.exe has been moved to the top of the list of scripts and has no parameters.



There are some cases where an end user is unable to participate with any of your automation processes. Traveling users and users that work from home are some common examples. For any user that will fall out of scope of your automation process, you must determine how you will handle those users. Since the only requirement is that the application execute on the user's session, you can often just direct the user to a network share and have them run the application manually (by double-clicking the app). Some other options may be to package the exe, config, and data file together in a zip file and have it available for download from an intranet web site.

Simple Deployment

The simplest deployment strategy for the Profile Manager is to have a network share that hosts the application, config, and data file. Consider a path as follows: \\server\share\profilemanager.

The user account that runs the migration application will need full control over the directory. When the migrator executes and completes a mailbox, it will create the data file in that location, or if the file already exists, will append to it. Obviously, to create or append to the file, the migrator account needs sufficient permission to the directory.

When an end user runs Profile Manager, he/she needs read access to the same location. Profile Manager will be running under the user's context, and as such the user needs to be able to execute the application from the directory (there is no need to copy the file locally) and to read the contents of the configuration and data files.

Additionally, Profile Manager will attempt to create and update a log file for each user that runs the application. The location of the logs is determined by a setting in the configuration file. Users therefore need sufficient permission to the logs directory to be able to create and modify files within it.

So, in summary:

- A network location to host the application, configuration, and data file

- Full control of the location for the migration service account
- Read access and execute for end users on both the shares permission and well as any NTFS permissions.
- Read+write+create permissions for end users to the logs folder
- A logon script (or other automation) to initiate the profile update process without user interaction and without the need to engage IT support to visit user desktops

Complex Deployment

A complex deployment is also supported for cases where there are multiple categories of end users. Categories of users, in this context, refer to users that need special handling with regards to their Outlook profiles. For instance, in some organizations some users connect to the Exchange Server via Outlook Anywhere (see: Outlook Anywhere) while others connect directly to the server over the LAN. Here we have 2 categories of user, however the simple deployment does not address this. The simple deployment has a single configuration file that is used for all users.

To support a complex environment, you can deploy multiple copies of Profile Manager and its config and data files. Conceptually you would have one location for each unique configuration, for example:

- \\server\share\standardUsers
    - The configuration file is set with default values for normal profiles
    - The data file only contains users that need this configuration
        - This subsequently means that this group of users will go through the migration process as a separate batch
- \\server\share\OutlookAnywhereUsers
    - The configuration file is set with values for Outlook Anywhere
    - The data file only contains users that need to use Outlook Anywhere
        - This subsequently means that this group of users will go through the migration process as a separate batch

Your logon script would then cause each ProfileManager.exe application to run during logon. This is safe because a particular mailbox will only exist in one of the 2 data files. If the user is an Outlook Anywhere user, their information will be in the data file in the \\server\share\OutlookAnywhereUsers location and not in the other.

This pattern can be expanded for as many different types of configurations you need to support. Note that for each separate configuration, you will also need to identify and isolate those users into groups for selection during the migration. Each mailbox (when successfully completed) in a batch of users to be migrated is written to a single, specified data file. As, such grouping of users for specific Outlook profile settings impacts how you select them for migration.

Options & Configuration

Profile manager has several configurable options that should be reviewed in relation to your migration. Although not every option is a necessity, you may find upon review that one or more of the options provide a better post-migration experience than currently being discussed. For instance, in many older environments users must create a VPN connection prior to connecting to their mailbox with Outlook. Often these are remote or traveling users (and as such are often revenue generating individuals) that need to work with Outlook. With the introduction of Outlook Anywhere, users can now securely access their mailbox without the VPN requirement. This Microsoft feature can sometimes have a considerably positive financial impact on the business.

Note: This is not an exhaustive list of possible settings and Administrators should use the profile configuration wizard that is part of the Mailbox Migration Manager application to assist with generating a properly formed config file.

The configuration file is a standard INI file layout. Each section and setting is described below. These settings are stored in the `prof.config` file.

Priasoft provides a utility to configure most settings in a config file. The Mailbox Migration Wizard automatically presents the utility –Profile Manager Configuration Editor – when a config file is not present in the specific location. You can manually run this utility at any time; the application is found in the Priasoft application directory (typically <Program Files>\Priasoft\Migration Suite for Exchange 6.0\) and is called ProfConfig.exe.

```
[Profile Update Manager]
puffile=\\psdc02-mesa\NETLOGON\profilemgr\mbm.puf
showdialog=YES
renameProfileAfterUpdate=YES
cachedmode=PRESERVEEXISTING
cachedModeDownloadOption=Default
cachedModePFFavs=Default
UseCurrentOSTPathFirst=YES
#CachedModeOSTPath=<Option is commented out and will use a default location under user's non-roaming profile>
enableRPCEncryption=YES

[Reports and Logs]
logpath=\\psdc02-mesa\NETLOGON\profilemgr\logs


[ROH Config]
```

A typical Outlook 2000 Profile Manager Config file.

[Profile Update Manager]

This INI section contains settings that relate to the general behavior of profilemanager.exe, not all possible settings are documented in the following list:

puffile

Value: name or path and name of the data file
Example: `puffile=\\server\share\folder\mbm.puf`

This setting describes the location of the data file that PROFILE MANAGER should use. Only mailboxes that appear in the data file will be updated, provided that they match with information in a user's Outlook profile.

The .PUF file is a text file that contains one line of text for each migrated mailbox. The ProfileManager searches the contents of this file for matching information from a user's Outlook profile, and upon a match, updates the profile.

This file is managed by the migration process and is updated after each mailbox completes.

If you need to deploy PROFILE MANAGER to non-local users – perhaps by using a zip file – you should set the value to the data file name only so that PROFILE MANAGER will look in its working directory for the data file. For instance: `puffile=mbm.puf`

Showdialog

Value: YES or NO
Example: `showdialog=YES`

This controls whether the ProfileManager shows a window on the user's desktop while it is running. It its recommended that this option be enabled because it provides administrators with a mechanism to determine if an automated process that is responsible for running ProfileManager is actually executing. Without a window, administrators cannot ask users if the application is running.

There are 2 options for showing a window: a simple window with custom text, or an uncompressed windows bitmap (.bmp). Only ONE of the 2 options can be used at a time.

dialogText

Value: some custom text that is less than 256 characters
Example: `dialogText=Please wait while your Outlook profile(s) are analyzed… This will only take a few seconds.`

If this setting is missing and ShowDialog = YES, the following default text will be shown: "Please wait while your profile(s) are updated..."

dialogBitmap

Value: name or path to an uncompressed bitmap file
Example: `dialogBitmap=\\server\shared\folder\bitmapFile.bmp`

The bitmap option points to an existing bitmap file in the file system. The fully qualifed path and file name needs to be specified and the bitmap needs to be readable by the end user that runs the ProfileManager.

renameProfileAfterUpdate

Value: YES or NO
Example: `renameProfileAfterUpdate=YES`

When this option is enabled, ProfileManager will rename the updated profile by appending a period (.) to the end of the profile name. The purpose of this option is to disconnect any existing Outlook Nickname Cache (see: About Autocomplete and NK2 File) file that is associated with the profile. By choosing to rename a profile in this way, users will receive a new nickname cache file. This prevents issues that occur due to a Cross-Forest migration.

Outlook's nickname cache contains addresses that are seen as a dropdown of previously used addresses when a user begins addressing a new message. Outlook, however, is unaware of Cross-Forest migrations and items in the cache that represent recipients from the prior Exchange organization **WILL NOT WORK** unless those recipients exist in the target Active Directory with a matching X500 address. Users will receive NDR's for any addresses used that represent recipients from the prior Exchange org.

If you are unable (or unwilling) to use Priasoft's directory synchronization tools, disconnecting the cache in this way is the only currently supported way of handling the cache file to prevent possible NDRs. Using the directory synchronization tools can help create a target address book that is fully populated with objects from the source environment and for which have matching X500 values; this will allow existing cached entries to be resolvable and will prevent NDRs.

If you choose to rename the Outlook profile users will start over with a new cache. As such, it may be prudent to communicate this to end users prior to the migration. Additionally, some users may complain that "all my contacts have been deleted!". This is because for some users, the nickname cache, in their mind, is the same as their contacts. The user will say that many of the smtp addresses they use are only in the nickname cache and they need them again, post-migration. This case is easy to handle by renaming the profile again by removing the period at the end of the profile name. When the user starts outlook again, they will have access to their original cache and can access their smtp addresses. They should then be instructed to create real contacts for those addresses and then afterwards rename the profile once again, adding the period to the end of the profile name.

cachedMode

Value: PreserveExisting | ForceOn | ForceOff | ForceRemove
Example: `cachedMode=PreserveExisting`

ProfileManager can set/reset/clear Outlook's 'Cached Mode' setting as part of the profile update process.

In many cases this is more efficient and reliable that using a group policy template for the fact that no control over a specific profile is provided with the group policy template. It is important to note that group policies that interact with Outlook should only be enabled AFTER ProfileManager has completed its updates. There are conflicts with profile access between Microsoft template and the profile update process.

Profile manager will only handle the cached mode setting if the profile is updated, and will only change the specific profile that is updated. Furthermore, ProfileManager provides options for Cached Mode that are not available by any other means.

Four options are provided to handle all situations:

- PreserveExisting
  - Current setting is maintained. Users with cached mode before migration will have cached mode after. User without, will not have cached mode
- ForceOn
  - the profile will be set for cached mode upon update, regardless of current settings
- ForceOff
  - the profile will be disabled for cached mode. This option only unchecks the box that enables cached mode.
  - In most cases this option should be avoided.
  - If the intent is to disable cached mode, use 'ForceRemove'
- ForceRemove
  - Unlike 'ForceOff', this setting will remove all settings for cached mode, which are of PRIME IMPORTANCE if cached mode should be enabled at some time after the profile is updated.
  - Since ForceOff does not reset the cached mode settings, Outlook will still have a reference to the old OST file which cannot be used when the mailbox exists in a different Exchange Org.

Two additional options exist for cached mode profiles:

cachedModeDownloadOption
Value: Default | Full | HeadersOnly | HeadersThenFull
Example: `cachedModeDownloadOption=HeadersOnly`

This is an Outlook feature that is controllable by end users and has no dependency upon Priasoft. See Cached Exchange Mode Settings for details about the various settings. Note that Priasoft does not support the **On Slow Connections Download Headers Only** option at this time.

A valuable point to consider here with regards to these settings is the "day after experience". For users with large mailboxes, or for a large number of users that all have their profiles updated to cached mode, the user experience can be degraded for a period of time after the migration and update of their profile. The reason is that all of the users mail must be downloaded from the new mailbox into their OST file. Different versions of Outlook handle this differently with Outlook 2003 being the most severe by blocking the user from sending mail until the mailbox is downloaded. If hundreds, or thousands of users are all downloading mail in the same window of time the day after the migration, this can put a strain on the Exchange server and disk system which in turn slows down the download for all users. In the case of large mailboxes, those users will have to wait a considerable amount of time.

Using one of the option provided here, such as HeadersOnly, greatly reduces the amount of data that is initially downloaded and can take some of the strain off of the Exchange servers. Several days later, instruction can be sent to users (via email and a screenshot perhaps) telling them they can, if they wish, change their setting back to Full Message Download.

Lastly, in conjunction with this experience, it may be wise to use the Mailbox Migration feature to Backfill mailboxes. Doing so allows you to setup the migration with a limited number of days of mail for the "day after", which in turn means less mail to download. When the backfill process starts migrating the balance of the mailbox, those items will synchronize with the users' OST file as normal.

cachedModePFFavs
Value: Default | YES | NO
Example: `cachedModePFFavs=Default`

For information about this feature, see: Public Folders and Cached Mode.

useCurrentOSTPathFirst
Value: YES | NO
Example: `useCurrentOSTPathFirst=YES`

When cached mode is enabled, Outlook allows for the setting of the location of the Offline STore file (OST). By default this location is in the user's non-roaming profile location (meaning on a local disk), however, if desired, a different setting can be applied.

When this option is enabled by use YES, ProfileManager will use the same location as any current OST file when it specifies the new OST file. If there is no current OST file configured, ProfileManager will use either a default setting (under the user's non-roaming profile), or a specific location as defined by the setting `CachedModeOSTPath`.

When this option is NO, ProfileManager will not evaluate any current OST file path and will use the setting of `CachedModeOSTPath`.

cachedModeOSTPath
Value: APPDATA | LOCALAPPDATA | <a custom location>
Example:
```
cachedModeOSTPath=LOCALAPPDATA
cachedModeOSTPath=d:\%username%\OSTFiles
```

If this setting is missing or commented out, the OST file will default to LOCALAPPDATA.

APPDATA defaults to the following location relative to the current user: %APPDATA%\Microsoft\Outlook
LOCALAPPDATA defaults to the following location relative to the current user: %LOCALAPPDATA%\Microsoft\Outlook

For custom locations, any text surrounded by %percent% signs will be evaluated as an environment variable and will be replaced with the appropriate value when it runs.

Note that it is NOT recommended to specify a location on a network drive or outside of the user's windows profile. Outlook needs read, write, and create privileges to the OST file path and placing the file outside of the user's windows profile may place it in a read only location. In any case where you use a custom location, you should validate that users have read+write+create permission to the location.

Also note that if a roaming profile location is specified - like %appdata% - the user's OST file will be copied to their home drive on the network each time they log off their computer. This may or may not be desired.

A common reason for the use of the option is to place the user's OST file in a location that is in scope of desktop backup procedures. Some customers instruct users that any files stored on specific Drive Letter or folder are backed up. It is wise to include a user's OST file in such a backup routine, however be aware that the OST file will typically equal the size of their mailbox – which may be several hundred megabytes, if not gigabytes, in size.

pstReconnectLoc
Value: <file system location where PST files are located>
Example: `pstReconnectLoc=\\server\share\folder_of_pst_files`

PST Reconnect is used for cases when a PST file exists prior to a profile update that should be **added** to a user's profile.

NOTE: Any PST file that is already attached to a profile prior to the migration will remain. This feature does not affect existing PST files that are already being used by end users.

A common scenario is one in which some or all of a user's mailbox was exported to a PST file prior to a migration and for which it is desired, through an automated process, to reconnect that PST file to a user's profile.

ProfileManager, upon update of a profile, can add an existing PST file to the profile, optionally copying the PST file locally to the user's computer before attaching.

ProfileManager looks in the supplied path for the first PST file that matches the user's primary SMTP address. This is to say that if the primary SMTP address for Joe User is joe.user@domain.com, ProfileManager will look for a file named joe.user@domain.com.pst in the supplied path.

If a file is found, it is added to the user's profile, or if enabled by the setting `pstReconnectCopyLocal`, is copied to the users computer (%UserProfile%\Application Data\Outlook) and is then added to the Outlook profile.

pstReconnectCopyLocal
Value: YES | NO
Example: `pstReconnectCopyLocal=YES`

When set to YES, a matching PST file is copied local before being added to the user's Outlook profile.

It is highly recommended to use this option as PST files over the LAN create unnecessary amounts of data traffic on the network. See this article for more information: http://support.microsoft.com/kb/297019

enableRPCEncryption
Value: YES | NO | Default
Example: `enableRPCEncryption=YES`

RPC Encryption is used to encrypt communication between Outlook and the Exchange server.

With the release of Exchange 2010 (and probably any release afterwards), the default setting on exchange servers REQUIRES RPC encryption in order to connect.

Prior versions of exchange (Ex2007 -> Ex5.5) have a default setting of ALLOW RPC encryption, meaning that if a client requests encryption, the server will oblige, otherwise it will allow un-encrypted traffic.

Of particular note, starting with Outlook 2007, the RPC Encrypt flag is enabled on new profiles, however, all previous clients (Outlook 2003 and earlier), do not set this flag when a profile is created. Furthermore, upgrades to Outlook (such as moving from OL2003 to OL2007 or OL2010) do not update any existing profiles. This means that in most cases Outlook profiles for 2007 and 2010 will still have a setting that does NOT require encryption.

In short, this means that if you have Outlook clients of version 2003 (note that Microsoft does not support Outlook 2000 or earlier on Exchange 2010) or have upgrade from Outlook 2003, and you want users to connect to an Exchange 2010 server, those profiles MUST have the RPC Encrypt flag enabled.

Priasoft highly recommends enabling this option IN ALL CASES. Even if you are not migrating to Exchange 2010 with the current project, we recommend enabling this setting. There will likely be a case in the future where the organization will upgrade to Exchange 2010 (or some later version). Setting the encryption flag now ensure that you do not have future issues of which you would be forced to visit user desktops to repair.

servicesToRemove

Value: servicename1[;servicename2[;..servicenameN]]
Example: `servicesToRemove=LiveMeeting;MSPST MS`

This setting allows you to specify one or more MAPI services (see:) that should be removed during the profile update process.

This may be valuable if you have a 3rd party service that you are not migrating forward to the new environment and for which users will not have access to after the migration completes. LiveMeeting is a common example. If you have OCS (see: Message Services) and have users that connect to OCS today, but post-migration OCS will be unavailable or will be available on a new separate system (for which the current LiveMeeting service is not configured), this feature will remove that service so that users are not faced with a non-working feature in Outlook.

Note that ALL instance of a service are removed, meaning that MSPST MS will remove ALL PST files from a user's Outlook Profile (the PST files themselves will remain on disk however).

Some common MAPI service names are:

- LiveMeeting – The Microsoft LiveMeeting Service
- CONTAB – The Outlook Address Book Service
- MSEMS – The Microsoft Exchange Service (this is the service that connects to Exchange mailboxes)
- MSPST MS – The ANSI (2gb max size) PST file service
- MSUPST MS – The UNICODE (20gb max size) PST file service

[Reports and Logs]

This configuration sections contains settings for reports and logs that can be generated during the update of an outlook profile.

logPath

Value: <a folder name or path where log files are created or updated>
Example: `logpath=\\server\share\folder\logs`

The Logs folder is a file system folder that is used to store detailed logs of ProfileManager's activities during execution.

You must make sure that the user context under which ProfileManager runs has Read/Write/Create access to the folder. ProfileManager will be executing as the end user (thus, with user's credentials) and so the Logs folder will need appropriate permissions to allow logging. It is common to give the Domain Users, Authenticated Users, or Everyone group Full Control on the logs folder.

If the logs folder is not supplied, is unreachable, is not writeable, or has any other issue, log files will be created in the user's desktop profile (%UserProfile%\Profile.Logs).

Log files created in the Logs folder have a format of "USERNAME@DOMAINNAME on COMPUTERNAME.profileupdate.log"

In addition to the log file, a zero-byte file is created that represents the result of a profile update operation. The format of the file is:

idx.ComputerName.SUCCESS.ProfileName.Username
OR
idx.ComputerName.FAILED.ProfileName.Username

This format is used by the Profile Update Monitor application in order provide a snapshot view of status. These files should not be moved or deleted.

getMailboxDetails

Value: 0 | 1
Example: `getMailboxDetails=1`

ProfileManager can gather and set simple mailbox details prior to a migration. Often it is desirable to know the size and item count of a mailbox prior to a migration. When this option is enabled, ProfileManager will attempt to update profiles first, then run in a hidden and idle state until Outlook is started by an end user.

Once Outlook has connected to the mailbox, ProfileManager will quickly and silently collect the mailbox size and item count and store these values on the user's directory object. No special permission are required for this operation to work. Users, by default already have appropriate permission as the 'owner' of the object.

Since the typical and recommended deployment of ProfileManager is before mailboxes are migrate, this feature can be used to gather details prior to the migration. Depending on how far in advance of the migration it is deployed, you may find value in capturing the metrics on a daily or weekly basis in order to track trends of mailboxes usage. Additionally, when the Priasoft Mailbox Migration Manager wizard later is run, and such a user is selected, the wizard will also grab the stored information from the directory and will provide the size (and overall batch size).

In Exchange 5.5, the value is stored in an LDAP property called Extension-Data. In all other exchange versions, this data is stored in the mailbox's directory object in the URL property.

getMailboxDetailsTimeout

Value: 0 to 1440
Example: getMailboxDetailsTimeout=15

The Timeout value above is used to control how long ProfileManager will wait for an Outlook session to start up. The value is in minutes and a value of zero (0) will wait indefinitely.

NOTE: The details gathering option causes the ProfileManager application to remain running after it's executed. Special consideration must be given to logon scripts and other automation processes. Commonly, logon scripts work sequentially – waiting for each script call to finish before moving to the next operation – and as such, unless ProfileManager is executed asynchronously, your logon script will appear to hang because it's waiting for ProfileManager to finish it's operation.

profileStatsLoc

Value: <a folder name or path to a folder at which profile details will be stored>
Example: `profileStatsLoc=\\server\share\folder`

This option, when a path is supplied, will enumerate all services and providers (see: Services and Providers) for each Outlook profile for the current user and will log detailed data for each. This information can be used to analyze the state of your Outlook profiles prior to the migration. You will be able to see which users have PST files, cached mode, and other detailed settings. The use of the information may help in determining how to group users for the migration.

The resulting file will have the format of user@computer.profileDetails.txt. All profiles for the current user will be analyzed and stored in this file. Note that subsequent executes of the Profile Manager will overwrite the previous, as such the current user must have read+write+create privilege to the configured location.

[ROH Config]

This section is used to configure an Outlook profile for remote access to an Exchange server using a Microsoft feature called RPC-over-HTTPS, also known as Outlook Anywhere. This feature allows users to remotely and securely connect to Exchange without the necessity of a VPN tunnel. More information can be reviewed here: RPC-over-HTTPS and Outlook Anywhere.

The option to configure ROH (short for RPC-over-HTTPS) during the profile update process is valuable because often the only time you want users to use this feature is when they are supposed to use their new mailbox. Microsoft's Group Policy templates do not allow you to specify when to make these changes and do not allow you to be specific with regards to a profile that should receive the settings. As mentioned earlier in this document, there are conflicts between Profile Manager and Microsoft Group Policy templates and such should not be enabled simultaneously.

ROHUpdateMode
Value: FORCEROH | FORCESTANDARD | ONLYIFROH | SKIPIFROH
Example: `ROHUpdateMode=FORCEROH`

- ForceROH
  - All matching profiles will be configured to use ROH settings.
  - This will only occur if there is a match between the primary mailbox in an Outlook profile and information in the data file.
- ForceStandard
  - All matching profiles will be configured to use standard MAPI connections.
- OnlyIfROH
  - Other updates will occur on the profile, however the settings for ROH will only be applied if the profile is already using ROH settings.
- SkipIfROH
  - Other updates will occur on the profile, however if the current profile is configured for ROH those settings will remain as-is.

The following values all refer to standard settings that are part of ROH. Please refer to Microsoft's documentation regarding specific meaning of the values.

UseRPConHTTPS
Value: YES | NO
Example: `UseRPConHTTPS=YES`

Reference: Connect to Exchange server using HTTP

ExchangeProxyServer
Value: the fully qualified domain name of your Exchange server
Example: `exchangeProxyServer=mail.contoso.com`

Reference: Configure RPC over HTTPS

UseSSLOnly
Value: YES | NO
Example: `UseSSLOnly=YES`

Reference: ROH Security

AuthenticatePrincipalName
Value: YES | NO
Example: AuthenticatePrincipalName=YES

Reference: Mutual Authentication

ExchangePrincipalName

Value: the FQDN of the server that is on the SSL cert, prefixed by the token: `msstd:`

Example: `exchangePrincipalName=msstd:exchangeserver.onsslcert.com`

Reference: Mutual Authentication

HttpFirstOnFast

Value: YES | NO

Example: `httpFirstOnFast=YES`

If Outlook determines it is on a fast network connection, it will use HTTPS first before trying to use RPC directly.

HttpFirstOnSlow

Value: YES | NO

Example: `httpFirstOnSlow=YES`

If Outlook determines it is on a slow network connection, it will use HTTPS first before trying to use RPC directly.

ProxyAuthStyle

Value: NTLM | Basic

Example: `proxyAuthStyle=NTLM`

Reference: Use Outlook Anywhere to connect to your Exchange server without a VPN

IMPORTANT: If you use Basic Authentication or NTLM Authentication and an LM Compatibility Level of less than 2, you will be prompted for a password each time a connection is made to Exchange. With Basic Authentication, the password is sent in clear text. For increased security, we recommend that you select the NTLM Authentication and Connect using SSL only options.

If the LM Compatibility Level is 2 or greater, the user can store the credential in Window's Credential Cache after which the user will not be prompted again unless the password changes.