Windows 10 Upgrade Documentation

Prerequisites

The clients on which an upgrade is to be performed must meet the following Windows 10 Minimum Prerequisites.
The minimum requirements for Windows 10 Release 2004 are:

- RAM: 2GB of RAM for 64 bit and 1GB of RAM for 32 bit computers.
- Storage: 20GB of free space on 64 bit and 16GB of free space on 32 bit systems.
- Screen resolution: 800×600
- Graphics: Microsoft DirectX 9 or later with WDDM 1.0
- Intel CPU: All processors up 10th-gen, Intel Xeon E-22xx, Atom, Celeron, and Pentium.
- AMD CPU: All processors up to seventh-gen.
- Qualcomm CPU: Snapdragon 850 and 8cx.

The client must have an upgradeable operating system installed. The Windows 10 Upgrade Matrix provides an overview of the possible upgrade paths in this case.

Preparations

Download the installation media

In order to upgrade to a newer release, you need an appropriate installation medium. The following details must be observed:

1. The installation media must be downloaded for the correct architecture (32bit/64bit).
2. The installation media must be downloaded for the correct Windows edition (Home/Pro/Education/Enterprise).
3. The installation media must be downloaded for the correct language.
4. The installation media must be downloaded for the correct Release ID (2004/20H2).

Installation media can be obtained from various sources:

- Microsoft Windows 10 Disc Image Download
- Media Creation Tool
- Volume Licensing Center
- MSDN
- Windows Insider Website
- Website for Education Editions (Product Key required)

Integration of the installation media

To obtain the required files, the downloaded ISO file must be unpacked. The unpacked files must be copied to the localsetup\installfiles folder of the windows10-upgrade package. After that a `sudo`

`opsi-set-rights` should be executed. Multiple installation media can also be included in the package, for example to upgrade to different release IDs, or to cover different architectures or languages. The following should be taken into account:

- The folder name must always start with installfiles!
  * To use the product property `automode` the release ID must be part of the folder name

Examples:

- `installfiles2004`
- `installfiles-20h2`
- `installfiles_21h1-32Bit` (from version 20.09-3)
- `installfiles_21h2-en` (from version 20.09-3)

In order for the package to recognize the different Installfiles folders, it must be reinstalled via `opsi-package-manager -i` after the folders have been created and loaded, due to the fact that integration of the Installfiles folders is done in the preinst/postinst scripts. The installfiles directory to be input can then be selected via the `installfiles_dir` product property.

Checking the installation media

With the tools `wiminfo` or `wimlib` the images contained in the install.wim can be read out. This can be used to determine the different editions, languages and architecture contained in the installation medium.

wiminfo

The tool is provided on Linux through the server package `opsi-windows-support`.
With `wiminfo` the information can be read out directly on the console of the OPSI server.
Example command to read the information:

- `wiminfo /var/lib/opsi/depot/windows10-upgrade/localsetup/installfiles/sources/install.wim`

wimlib

The tool is located in the package under the following paths:

- `localsetup\wimlib\x64\wimlib-imagex.exe`
- `localsetup\wimlib\x86\wimlib-imagex.exe.\`

Example command to read the information (\\<opsi-server>\opsi_depot share mounted as drive p:):

- `p:\windows10-upgrade\localsetup\wimlib\x64\wimlib-imagex.exe" info "p:\windows10-upgrade\localsetup\installfiles2009\sources\install.wim"`.

During the upgrade, the image contained in the directory selected by product property `installfiles_dir` is read with `wimlib` and the output is stored in the logfile.

Symlinks

As of version 21h1-1, the installfiles folders do not have to be stored directly in the localfiles folder, but can also be included via symlink. Example for creating a symlink to the installfiles folder from the netboot package win10-x64:

- `cd /var/lib/opsi/depot/windows10-upgrade/localsetup`
- `ln -s ../../win10-x64/installfiles/ .`

After creating the symlink, the package must be installed again via `opsi-package-manager -i`, as the integration of the installfiles folders is done in the preinst/postinst scripts. The installfiles directory to be used can then be selected via the product property `installfiles_dir`.

Product Properties

- `automode`
  - False (Default)
    - The release ID to which the upgrade is to be performed must be selected via the product property `upgrade_to_version`.
  - True
    - An attempt is made to read the release ID from the folder name of the folder selected via the product property `installfiles_dir`.
    - Example: If the folder installfiles2004 was selected in the product property `installfiles_dir`, 2004 will be used as the release ID. The Release ID selected under `upgrade_to_version` will be ignored in this case!
    - If no valid release ID can be read, the value selected under the product property `upgrade_to_version` is used as fallback.
- `bitlocker_suspend`
  - AlwaysSuspend (Default)
    - Always suspend BitLocker during upgrade.
  - ForceKeepActive
    - Enable upgrade without suspending BitLocker, but if upgrade doesn't work, fail the upgrade.
  - TryKeepActive
    - Enable upgrade without suspending BitLocker, but if upgrade doesn't work, then suspend BitLocker and complete the upgrade.
- `copy_files_locally`
  - False (Default)
    - The installation files are obtained from `opsi_depot` during the upgrade.
  - True
    - The installation files are copied locally to the folder `%SystemDrive%\opsi.org\usertmp\windows10-upgrade` before the upgrade.
- `debug`
  - False (Default)
    - Locks keyboard and mouse input during opsiSetupUser auto logon to avoid user interaction.
  - True
    - Keyboard and mouse remain active during auto logon for debugging in case of error.

- `delete_windows_old`
  - False (Default)
    - The Windows.old folder remains after the upgrade.
  - True
    - After a successful upgrade, the Windows.old folder is deleted.
- `delete_windows_update_cache`
  - False (Default)
    - The Windows update cache in the folder `SystemRoot\SoftwareDistribution` will be kept.
  - True
    - The Windows Update cache in the folder `SystemRoot\SoftwareDistribution` will be cleared before the upgrade.
- `dynamic_update`
  - Disable (Default)
    - Dynamic Update operations are not performed.
  - Enable
    - Dynamic Update operations are performed.
  - NoDrivers
    - Dynamic Update operations are performed except driver acquisition.
  - NoDriversNoLCU
    - Dynamic Update operations are performed except driver and latest cumulative update acquisition.
  - NoLCU
    - Dynamic Update operations are performed except latest cumulative update acquisition.
- `encryption_driver`
  - Specifies a path containing drivers for third party encryption software.
- `execution_method`
  - loginOpsiSetupUser
    - The opsiclientd service creates the user opsiSetupUser and logs it in.
    - The explorer.exe is started as shell.
    - The desktop of the opsiSetupUser is visible.
    - After logging in, the upgrade starts after a short delay.
    - The installation files can be retrieved from the server, or stored locally on the client (see `copy_files_locally`).
  - runAsOpsiSetupUser
    - The opsiclientd service creates the user opsiSetupUser and logs it in.
    - The shell powershell.exe is started and triggers the local opsi-script, which executes the upgrade.
    - The desktop of the opsiSetupUser is not visible.
    - The installation files must be saved locally on the client.
    - The value set under `copy_files_locally` is ignored and automatically set to true.
  - runOpsiScriptAsOpsiSetupUser (default)
    - The opsiclientd service creates the user opsiSetupUser and logs it in.
    - The powershell.exe is started as shell and triggers the `action_processor_starter.exe`, which establishes a service connection and then executes the upgrade.
    - The desktop of the opsiSetupUser is not visible.
    - The installation files can be retrieved from the server, or stored locally on the client (see `copy_files_locally`).

    - If a client is in WAN/VPN mode (automatic detection), this product property is ignored and the installation is carried out with the following options: (See also chapter WAN/VPN Mode).
        - The opsiclientd service creates the user opsiSetupUser and logs it in.
        - The powershell.exe is started as shell and triggers the local opsi-script, which executes the upgrade.
        - The desktop of the opsiSetupUser is not visible.
        - The locally cached installation files are used.
- `image_index`
    - Empty (Default)
        - Instructs Windows Setup which OS image to install from install.wim if multiple images may be applicable.
- `installfiles_dir`
    - Selection of the folder containing the installation files.
    - The folder name must always begin with installfiles!
    - If the product property `automode` is to be used, the release ID must be appended to the folder name (e.g. installfiles2004).
    - To ensure that all installfiles directories contained in the package are listed in the product property `installfiles_dir`, the package may have to be installed again via opsi-package-manager -i after the folders have been created, as the property is filled via the preinst/postinst scripts.
- `language_from_language_pack` (See chapter `Language Packs`)
    - False (default)
        - Select if the client was not installed with an installation media that contains language packs.
    - True
        - Select if the client was installed with an installation media that contains Language Packs and the client language is provided by a Language Pack.
- `language_installation_media` (See chapter `Language Packs`)
    - Specify the language code of the original installation medium of the client.
    - Examples:
        - 0407 for German
        - 0409 for English
- `migrate_drivers`
    - All (Default)
        - Instructs Windows Setup to migrate all drivers from the existing installation during the upgrade.
    - None
        - Instructs Windows Setup not to migrate any drivers from the existing installation during the upgrade.
- `mode`
    - upgrade (default)
        - Upgrade to the selected release ID.
    - validate_before_upgrade
        - Trial run to determine compatibility problems. If no problems are detected the upgrade is started.
    - validate_only
        - Trial run to determine compatibility problems.
- `post_oobe`
    - Path to setupcomplete.cmd
    - More info at:

https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/add-a-custom-script-to-windows-setup?view=windows-11

- `productkey`
  - Licence key to be used for the installation.
- `proxy_ip`
  - Allows you to specify the IP of a proxy server.
- `proxy_port`
  - Allows you to specify the port number of a proxy server.
- `quiet`
  - False
    - The installation starts in interactive mode and requires the input of a user.
    - Useful to get a meaningful error message in case of an error.
  - True (Default)
    - The installation starts in silent mode without user interaction.
- `restore_health`
  - False (default)
    - „SFC /scannow“ and „DISM /online /cleanup-image /restorehealth“ are not executed before the upgrade.
  - True
    - „SFC /scannow“ and „DISM /online /cleanup-image /restorehealth“ will be executed before the upgrade.
- `setup_after_install`
  - Selection of OPSI products to be installed after a successful upgrade.
  - Selectable products (manually upgradeable):
    - config-win10
    - mshotfix
    - opsi-client-agent
- `showoobe`
  - Full
    - Users must click through the Out Of Box Experience (OOBE) dialog when logging in after the upgrade.
  - None (Default)
    - Skips the Out Of Box Experience (OOBE) dialog when a user logs in after the upgrade.
- `telemetry`
  - Disable (Default)
    - No telemetry data from the upgrade will be sent to Microsoft.
  - Enable
    - Telemetry data from the upgrade will be sent to Microsoft.
- `timeout_opsisetupuser`
  - 120 (Default)
    - Specifies the timeout value in minutes for opsiServiceCall*runAsOpsiSetupUser and opsiServiceCall*runOpsiScriptAsOpsiSetupUser
- `unset_after_install`
  - Selects the OPSI products to be set to not_installed after a successful upgrade.
- `upgrade_to_version`
  - Selection of the release ID to which the upgrade should be performed.
  - Ignored if `automode` is set to true.
    - Only if no valid release ID could be read via automode, the value selected here is used as fallback.
  - latest (Default)

- Always refers to the latest release ID that was current at the time the package was created.

WAN/VPN mode

The package automatically detects whether a client is in WAN/VPN mode. The locally cached installation files are used.

Special features

Important! After each change to the installfiles folders, the package must be reinstalled via `opsi-package-manager -i` so that the .files file is regenerated!
Packages selected via `setup_after_install` can only be cached when the client is able to contact the server successfully the next time. This usually happens only after a VPN connection has been established after a user has logged in.

Language Packs

Windows installations with installed language packs are a special case. In order to successfully update an installation that uses language packs, a few points must be observed.

Preparation of the installation medium

All required Language Packs must be included in the install.wim of the installation medium for the upgrade. Please note that the language packs must be downloaded according to the version of the installation medium. For example, Language Packs in version 1909 cannot be included in a 21H1 installation medium. Language Packs are integrated using the DISM tool. Instructions and further information can be found in the following links:

Add or Remove Packages Offline Using DISM
How to Add Language Packs Offline Using DISM
DISM Languages and International Servicing Command-Line Options
Available language packs for windows

Process

The operating system to be upgraded must be reset to the language of the original installation media before the upgrade can take place. For example, if a client was installed from English installation media containing a German language pack and German was configured as the client's primary language, the language must be set to English before upgrading. After the upgrade, the language can be set back to German if the installation media used during the upgrade has a German language pack integrated that matches its ReleaseID. This process has been integrated automatically in version 21h1-1 of the windows-upgrade packages. Before the upgrade, the current language is read and saved. Then the language is set to the language of the installation medium used to install the client.

After the upgrade, the language is set back to the values read out at the beginning.

Client Settings

- The product property `language_from_language_pack` must be set to true.
- The Product Property `language_installation_media` must be set to the four-digit Language Code of the installation medium.
    - Examples:
        - 0407 for German
        - 0409 for English

Upgrade from Windows 10 previous versions

The following Windows 10 previous versions can be upgraded to Windows 10:

- Windows 7
- Windows 8
- Windows 8.1

See also: Windows Upgrade Matrix
Before upgrading, all current Windows updates should be applied.

Debugging

Several circumstances can cause an upgrade to fail:

- Client does not meet the minimum requirements for Windows 10.
- System Reserved Partition too small
- Virus scanner prevents the upgrade
- Incompatible drivers on the client
- Incompatible software on the client
- Wrong installation media
    - Wrong architecture
    - Wrong language
    - Wrong edition
- Activation status of the operating system before the upgrade
- User interaction during the upgrade (e.g. restarting or switching off the client)

Evaluation of the exit code

As of version 20.09-1, the exit codes of the upgrade are evaluated in more detail and a meaningful error message with tips on how to fix it is written in the log file. However, the list of exit codes is certainly not complete, as there is no complete listing of all exit codes on the part of Microsoft.

Included upgrade log files

The following upgrade logfiles are included in the logfile of the package in case of an error (if available) and can be analysed conveniently in the Configed:

- „%opsiLogDir%\windows10-upgrade.log\Panther*APPRAISER*.xml“
- „%opsiLogDir%\windows10-upgrade.log\Panther*CompatData*.xml“
- „%opsiLogDir%\windows10-upgrade.log\MoSetup\ActionList.xml“
- „%opsiLogDir%\windows10-upgrade.log\MoSetup\Bluebox.log“
- „%opsiLogDir%\windows10-upgrade.log\MoSetup\DeviceInventory.xml“
- „%SystemRoot%\Logs\SetupDiag\setupdiagresults.xml“
- „%opsiLogDir%\opsisetupuser.log“
- „%opsiLogDir%\SetupDiagResults.log“
- „%opsiLogDir%\windows10-upgrade.log\Panther\diagerr.xml“
- „%opsiLogDir%\windows10-upgrade.log\Panther\MigLog.xml“
- „%opsiLogDir%\windows10-upgrade.log\Panther\ScanResult.xml“
- „%opsiLogDir%\windows10-upgrade.log\Panther\setupact.log“
- „%opsiLogDir%\windows10-upgrade.log\Panther\setuperr.log“
- „%opsiLogDir%\sfcdetails.txt“
- „%SystemRoot%\logs\dism\dism.log“

Solutions to problems with the Windows Upgrade

To find incompatible software and drivers look in the windows10-upgrade logfile, or directly in „%SystemDrive%\$WINDOWS.~BT\Sources\Panther\scanresult.xml“ and %SystemDrive%\$WINDOWS.~BT\Sources\Panther\Compatdata*.xml (several individual log files) for the following strings:

- `CompatibilityInfo BlockingType="Hard"`
- `CompatibilityInfo BlockMigration="True"`

In case of an incompatible driver, the corresponding .inf file should be displayed (e.g. oem11.inf). The .inf files are located in the folder `%SystemDrive%\Windows\INF`. Open them with a text editor to find the driver name. Uninstall the driver(s) found on the client and restart the upgrade.

Important! Two drivers with the status `CompatibilityInfo BlockMigration="True"` are **always** found (depending on the system in different oemxx.inf files) The associated devices are:

- `Microsoft Print to PDF`
- `Microsoft XPS Document writer`

Although these two drivers are displayed with `CompatibilityInfo BlockMigration="True"`, they **do not** prevent the upgrade and can be ignored.

To get a meaningful error message you can start the upgrade with the product properties `quiet` = false and `debug` = true. **Important!** This will start the upgrade in interactive mode and require user input! During the upgrade process you will **mostly** get a more detailed error message in the GUI.

The upgrade can be started with the product property `mode` = validate or `mode` = validate_before_upgrade. In this case, only a test run of the upgrade is started. Any errors that may

occur can thus be determined before the actual upgrade.

Solutions for problems with the windows10-upgrade package

Unfortunately, problems with the windows10-upgrade package itself can occur from time to time. In general, it is recommended to always use the latest version of the windows10-upgrade package!

Auto Logon problems

There can be problems with the Auto Logon of the opsiSetupUsers. Usually group policies prevent the temporary opsiSetupUser from logging in. In this case you should check if the logon is only allowed for certain local users or if the password policy prevents the login.

Windows dialogues that require user interaction at login can also prevent automatic login. The windows10-upgrade package suppresses all known dialogues, but depending on the release ID, new dialogues can always be added that prevent logging in. In this case, a screenshot of the corresponding dialogue is always required so that a workaround for this can be included in the next version of the windows10-upgrade package.